Multi-solver algorithms for the partitioned simulation of fluid-structure interaction

Joris Degroote[a,], Jan Vierendeels[a]

[a]*Department of Flow, Heat and Combustion Mechanics, Ghent University, Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium*

Abstract

In partitioned fluid-structure interaction simulations, the flow equations and the structural equations are solved separately. As a result, a coupling algorithm is needed to enforce the equilibrium on the fluid-structure interface in cases with strong interaction. This coupling algorithm performs coupling iterations between the solver of the flow equations and the solver of the structural equations. Current coupling algorithms couple one flow solver with one structural solver. Here, a new class of multi-solver quasi-Newton coupling algorithms for unsteady fluid-structure interaction simulations is presented. More than one flow solver and more than one structural solver are used for a single simulation. The numerical experiments demonstrate that the duration of a simulation decreases as the number of solvers is increased.

Keywords: fluid-structure interaction, partitioned, coupling algorithm, quasi-Newton, tube, cluster

1. Introduction

The mutual interaction between a fluid flow and a deformable structure is referred to as fluid-structure interaction (FSI). Life-saving examples are the opening of a parachute [1] or an air bag [2]. Undesired occurrances of fluid-structure interaction are collapses of bridges [3] and cooling towers due to wind [4], as well as flutter of aircraft wings [5] and turbine blades [6]. In the biomedical field, the

Email addresses: Joris.Degroote@UGent.be (Joris Degroote), Jan.Vierendeels@UGent.be (Jan Vierendeels)

URL: http://www.fsi.ugent.be/ (Joris Degroote)

interaction between an elastic artery [7, 8] or a heart chamber [9] and the blood that flows through them is of interest. Also for the design of artificial heart valves [10–12], fluid-structure interaction needs to be taken into account. As a result of these numerous applications and the increase in computational power, the numerical simulation of fluid-structure interaction has gained interest.

For the simulation of fluid-structure interaction, there are two approaches. Monolithic simulation techniques solve the governing equations of the fluid flow and of the structural deformation simultaneously [13–16]. Conversely, partitioned techniques solve the flow equations and the structural equations separately. An advantage of the partitioned approach to simulating this coupled problem is that the flow equations and the structural equations can be solved with a different solution technique. Moreover, the partitioned approach allows to reuse existing flow solvers and structural solvers.

In this article, the focus lies on partitioned simulation techniques which couple the flow solver and the structural solver as 'black boxes', which means that the discretization and solution techniques of the solvers do not have to be known. The partitioned techniques can be categorized as explicit or implicit. Several explicit (also known as loosely or weakly coupled) partitioned techniques exist [5, 17–20], which are suitable for aeroelastic simulations [21]. These techniques solve the flow equations and the structural equations separately and only once (or a fixed number of times) in each time step. Therefore, these techniques do not enforce the equilibrium of the stress and velocity (or displacement) on the fluid-structure interface, which results in restrictions on the time step for stability reasons [21–23].

Implicit (or strongly coupled) partitioned techniques enforce the equilibrium of the stress and velocity (or displacement) on the fluid-structure interface in each time step. This can be achieved by, for example, Gauss-Seidel iterations or Newton-Raphson iterations. During Gauss-Seidel iterations in a time step, the flow equations and the structural equations are solved successively until the user-defined convergence tolerance is reached. After the solution of the flow equations, the boundary condition on the solid side of the interface is updated and vice versa. Once these coupling iterations within the time step have converged, the solution is the same (up to the convergence tolerance) as would have been found with a monolithic solver. Several strongly coupled partitioned techniques are able to couple black-box solvers, for example Gauss-Seidel iterations with Aitken relaxation [24, 25], the Interface Generalized Minimal Residual method (Interface-GMRES) [26], the Interface Quasi-Newton technique with an approximation for the Inverse of the Jacobian from a Least-Squares model (IQN-ILS)

[27] and the Interface Block Quasi-Newton technique with an approximation for the Jacobians from Least-Squares models (IBQN-LS) [28]. Also Robin boundary conditions can accelerate the convergence of the coupling iterations [29].

Nowadays, fluid-structure interaction simulations are often performed on clusters for high-performance computing (HPC). These computers consist of a large number of cluster nodes, each containing a small number of multi-core processors and an amount of memory. Although fast interconnects between the cluster nodes exist, they are still slower than the communication lines inside the cluster nodes. By running the flow solver and the structural solver in parallel, i.e. on more than one core of the cluster, a fluid-structure interaction calculation can generally be accelerated. Optimally, the speed-up from parallelization would be linear: doubling the number of cores should halve the calculation's duration, and doubling it a second time should again halve the duration. However, very few parallelization algorithms achieve optimal speed-up. Most of them have a near-linear speed-up for small numbers of cores, which flattens out into a constant value or even decreases for large numbers of cores. The end of the near-linear speed-up depends on several factors, such as the number of degrees of freedom, the interconnect, the solvers, etc.

Fluid-structure interaction simulations are often unsteady calculations with a large number of time steps that have to be calculated consecutively and with a modest number of degrees of freedom to current standards. Due to this modest number of degrees of freedom, the near-linear speed-up can end at a relatively low number of cores. At that point, increasing the number of cores per solver no longer leads to a significant reduction of the calculation time. However, this article demonstrates that the calculation time can also be reduced by increasing the number of flow solvers and structural solvers, while keeping the number of cores per solver constant. Figure 1 illustrates this novel multi-solver approach to partitioned fluid-structure interaction simulations.

In this article, the new multi-solver approach is explained in detail and applied to the IQN-ILS [27] and IBQN-LS [28] coupling algorithms. These quasi-Newton algorithms construct least-squares models for the flow solver and structural solver, which are treated as black boxes. The least-squares models are built in each time step using the stress and displacement on the fluid-structure interface during the coupling iterations, as will be explained below. If consecutive time steps are sufficiently similar, data from previous time steps can be reused in the least-squares model of the current time step [30]. However, this reuse has to be applied with caution as the data from previous time steps is only approximately correct at the current time level [26, 31]. Therefore, the multi-solver quasi-Newton coupling al-

gorithms presented in this article first recalculate the data from the previous time steps at the current time level, before including that data in a least-squares model.

The remainder of this article is organized as follows. First, Section 2 gives a brief description of the governing equations and the notations. Then, Section 3 summarizes the IQN-ILS and IBQN-LS algorithms, as well as the construction of a least-squares model and the reuse of data from previous time steps. Subsequently, the detailed explanation of the new multi-solver algorithms is given in Section 4. The numerical results in Section 5 illustrate the performance of the multi-solver coupling algorithms. Finally, Section 6 offers the conclusions.

2. Governing equations

Figure 2 depicts an abstract fluid-structure interaction problem, with the subscripts f and s respectively denoting fluid and structure. The subdomains are indicated as Ω_f and Ω_s and their boundaries as Γ_f and Γ_s. The fluid-structure interface $\Gamma_i = \Gamma_f \cap \Gamma_s$ is the common boundary of these subdomains. $\vec{e}_x$, $\vec{e}_y$ and $\vec{e}_z$ are the unit vectors in the horizontal, vertical and out-of-plane direction, respectively.

2.1. Flow equations

The unsteady flow of a fluid is governed by the conservation of mass and the Navier-Stokes equations, given by

$$\frac{\partial \rho_f}{\partial t} + \nabla \cdot (\rho_f \vec{v}) = 0 \tag{1a}$$

$$\frac{\partial \rho_f \vec{v}}{\partial t} + \nabla \cdot (\rho_f \vec{v}\vec{v}) - \nabla \cdot \bar{\sigma}_f = \vec{f}_f \tag{1b}$$

for each point $\vec{x} \in \Omega_f$. In these equations, ρ_f is the fluid density, $\vec{v}$ the flow velocity and t the time. $\vec{f}_f$ represents the body forces per unit of volume on the fluid. For incompressible, Newtonian fluids with dynamic viscosity μ_f, the stress tensor $\bar{\sigma}_f$ is defined as

$$\bar{\sigma}_f = -p\bar{I} + 2\mu_f \bar{\epsilon}_f \tag{2a}$$

with p the pressure and $\bar{I}$ the unit tensor. The rate of strain tensor $\bar{\epsilon}_f$ is given by

$$\bar{\epsilon}_f = \frac{1}{2}\left[\nabla \vec{v} + (\nabla \vec{v})^{\mathrm{T}}\right]. \tag{2b}$$

2.2. Structural equations

The deformation $\vec{u}$ of the structure is determined by the conservation of momentum

$$\rho_s \frac{\mathrm{d}^2 \vec{u}}{\mathrm{d}t^2} - \nabla \cdot \bar{\sigma}_s = \vec{f_s} \tag{3}$$

for each point $\vec{x} \in \Omega_s$ with ρ_s the structural density, $\bar{\sigma}_s$ the Cauchy stress tensor and $\vec{f_s}$ the body forces per unit volume on the structure. In large displacement calculations, the relation between the second Piola-Kirchhoff stress tensor $\bar{S}_s$ and the Green-Lagrange strain tensor $\bar{E}_s$ is imposed by the constitutive equation of the material. The second Piola-Kirchhoff stress tensor combines forces in the reference configuration with areas in the reference configuration, whereas the Cauchy stress tensor combines forces in the deformed configuration with areas in the deformed configuration. The relation between these tensors is given by

$$\bar{S} = J\bar{F}^{-1}\bar{\sigma}_s\bar{F}^{-\mathrm{T}} \tag{4}$$

with $\bar{F}$ the deformation gradient tensor and $J = \det(\bar{F})$. The Green-Lagrange strain tensor for large displacements is given by

$$\bar{E}_s = \frac{1}{2}\left[\nabla\vec{u} + (\nabla\vec{u})^{\mathrm{T}} + (\nabla\vec{u})^{\mathrm{T}}\nabla\vec{u}\right]. \tag{5}$$

All displacements are relative to the initial (reference) geometry.

2.3. Equilibrium conditions

The equilibrium conditions on the fluid-structure interface ($\vec{x} \in \Gamma_i$) are the kinematic condition

$$\vec{v} = \frac{\mathrm{d}\vec{u}}{\mathrm{d}t} \tag{6}$$

and the dynamic condition

$$\bar{\sigma}_f \cdot \vec{n}_f = -\bar{\sigma}_s \cdot \vec{n}_s, \tag{7}$$

which stipulate that the velocity and the stress have to be the same on both sides of the interface. The vector $\vec{n}_{f,s}$ is the unit normal that points outwards from the domain $\Omega_{f,s}$. Appropriate boundary conditions are imposed on $\Gamma_f \setminus \Gamma_i$ and on $\Gamma_s \setminus \Gamma_i$, depending on the problem at hand. A Dirichlet-Neumann decomposition of the fluid-structure interaction problem is applied, so the flow equations are solved with a given velocity (or displacement) of the fluid-structure interface and the structural equations are solved with a given stress on the interface.

2.4. Discrete equations

The flow equations and the structural equations are discretized in space and time with a method of choice, which results in two coupled systems of discrete equations with the flow variables and the structural variables as unknowns. As the solvers are coupled as black boxes, the details of the discretization techniques do not have to be known. The discrete flow equations are represented by $\boldsymbol{F}$ and the discrete structural equations by $\boldsymbol{S}$. The vector $\boldsymbol{v}$ groups all flow variables (velocity, pressure, etc.) in Ω_f; the vector $\boldsymbol{u}$ groups all structural variables (displacement, stress, etc.) in Ω_s. The displacement of the interface Γ_i is represented by the vector $\boldsymbol{d}$ and the stress on the interface by the vector $\boldsymbol{s}$. In the case of a Dirichlet-Neumann decomposition, the displacement of the interface is considered as a function of the structural degrees of freedom ($\boldsymbol{d} = \boldsymbol{d}(\boldsymbol{u})$) and the stress on the interface as a function of the flow degrees of freedom ($\boldsymbol{s} = \boldsymbol{s}(\boldsymbol{v})$). Hence, the coupled problem can be written as

$$\begin{cases} \boldsymbol{F}(\boldsymbol{v}, \boldsymbol{d}(\boldsymbol{u})) = \boldsymbol{0} \\ \boldsymbol{S}(\boldsymbol{u}, \boldsymbol{s}(\boldsymbol{v})) = \boldsymbol{0} \end{cases} \tag{8}$$

in which all variables are at the new time level t^{n+1}; the dependence of the solution on the variables at $t^n, t^{n-1}, \ldots$ is hidden.

The flow solver calculates the flow variables $\boldsymbol{v}$ that satisfy $\boldsymbol{F}(\boldsymbol{v}, \boldsymbol{d}(\boldsymbol{u})) = \boldsymbol{0}$ for a given interface diplacement $\boldsymbol{d}$. From the flow field $\boldsymbol{v}$, the stress on the interface $\boldsymbol{s}$ is extracted. Therefore, the flow solver is represented by the function

$$\boldsymbol{s} = \boldsymbol{\mathcal{F}}(\boldsymbol{d}). \tag{9}$$

Similarly, the structural solver calculates the structural variables $\boldsymbol{u}$ that satisfy $\boldsymbol{S}(\boldsymbol{u}, \boldsymbol{s}(\boldsymbol{v})) = \boldsymbol{0}$ for a given stress on the interface $\boldsymbol{s}$. The displacement of the interface $\boldsymbol{d}$ is subsequently extracted from $\boldsymbol{u}$, so the structural solver is represented by

$$\boldsymbol{d} = \boldsymbol{\mathcal{S}}(\boldsymbol{s}). \tag{10}$$

The flow solver and the structural solver defined above are considered as black-box functions, which can be evaluated for a given value of their argument but whose Jacobian matrix is inaccessible. The argument and the return value of these functions are also referred to as input and output, respectively. The only requirements for the black-box flow solver are that it should be possible to impose a displacement of the interface and to extract the resulting stress distribution on the interface (vice-versa for the structural solver). If a solver does not satisfy these

requirements, it is not suitable for partitioned fluid-structure interaction calculations, regardless of the coupling algorithm.

With the above definitions, the equilibrium on the fluid-structure interface is given by

$$\boldsymbol{d} = \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}(\boldsymbol{d}). \tag{11}$$

If the flow equations and the structural equations are not discretized in the same way on the fluid-structure interface, there has to be an interpolation between the solvers. In this article, it is tacitly assumed that this interpolation is included in either the function $\boldsymbol{\mathcal{F}}$ or the function $\boldsymbol{\mathcal{S}}$ if necessary. An overview of interpolation techniques can be found in [32].

3. Quasi-Newton coupling algorithms

In the explanation of the qausi-Newton coupling algorithms, a prime denotes the Jacobian matrix of a function and a hat refers to an approximation. The output of a solver is indicated with a tilde as this value is not always directly given as input to another solver. All coupling algorithms begin a new time step with an extrapolation of the interface's displacement

$$\boldsymbol{d}^{n+1,0} = \frac{5}{2}\boldsymbol{d}^{n} - 2\boldsymbol{d}^{n-1} + \frac{1}{2}\boldsymbol{d}^{n-2}, \tag{12}$$

based on the previous time steps. This extrapolation is third-order accurate if the time step is constant; lower-order extrapolations are used for the first two time steps. In the notation $\boldsymbol{d}^{n+1,0}$, the first superscript refers to the time step. This superscript is further omitted if it is equal to $n+1$ as almost all variables are at this time level. The second superscript k (equal to 0 in this case) denotes the coupling iteration within time step $n+1$. Superscripts that are a number instead of a letter refer to a coupling iteration within time step $n+1$.

3.1. IQN-ILS

This section summarizes the IQN-ILS coupling algorithm with a matrix-free implementation of the least-squares model [27]. The fluid-structure interaction problem in Eq. (11) can be reformulated as a set of equations in the interface's displacement

$$\boldsymbol{\mathcal{R}}(\boldsymbol{d}) = \boldsymbol{0}, \tag{13}$$

with $\boldsymbol{\mathcal{R}}$ being the residual operator

$$\boldsymbol{\mathcal{R}}(\boldsymbol{d}) = \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}(\boldsymbol{d}) - \boldsymbol{d}. \tag{14}$$

This generally nonlinear set of equations can be solved by means of Newton-Raphson iterations

$$\text{solve} \quad \boldsymbol{\mathcal{R}}'^k \Delta \boldsymbol{d}^k = -\boldsymbol{r}^k \tag{15a}$$

$$\boldsymbol{d}^{k+1} = \boldsymbol{d}^k + \Delta \boldsymbol{d}^k \tag{15b}$$

with the residual calculated as

$$\boldsymbol{r}^k = \boldsymbol{\mathcal{R}}(\boldsymbol{d}^k) = \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}(\boldsymbol{d}^k) - \boldsymbol{d}^k = \tilde{\boldsymbol{d}}^k - \boldsymbol{d}^k. \tag{16}$$

In Eq. (15a), $\boldsymbol{\mathcal{R}}'^k$ denotes the Jacobian of $\boldsymbol{\mathcal{R}}$, evaluated at $\boldsymbol{d}^k$. The Newton-Raphson iterations in the time step have converged when $||\boldsymbol{r}^k||_2 \leq \varepsilon_o$ with ε_o the convergence tolerance. However, the exact Jacobian of $\boldsymbol{\mathcal{R}}$ is unknown as the Jacobians of $\boldsymbol{\mathcal{F}}$ and $\boldsymbol{\mathcal{S}}$ are unavailable. Moreover, the linear system in Eq. (15a) with as dimension the number of degrees of freedom in the displacement of the fluid-structure interface has to be solved in each Newton-Raphson iteration.

If the Jacobian $\boldsymbol{\mathcal{R}}'$ is approximated by applying the least-squares technique introduced by Vierendeels et al. [28], then quasi-Newton iterations are performed and black-box solvers can be used. However, the linear system in Eq. (15a) still needs to be solved. Therefore, it is more advantageous to approximate the *inverse* of the Jacobian. The quasi-Newton iterations with the approximation for the inverse of the Jacobian can be written as

$$\boldsymbol{d}^{k+1} = \boldsymbol{d}^k + \widehat{(\boldsymbol{\mathcal{R}}'^k)^{-1}} \left(-\boldsymbol{r}^k\right). \tag{17}$$

The complete IQN-ILS technique with reuse of data from q time steps is shown in Algorithm 1.

It can be seen from Eq. (17) that the approximation for the inverse of the Jacobian does not have to be created explicitly; a procedure to calculate the product of this matrix with the vector $-\boldsymbol{r}^k$ is sufficient. The vector $-\boldsymbol{r}^k$ is the difference between the desired residual, i.e. $\boldsymbol{0}$, and the current residual $\boldsymbol{r}^k$ and it is further denoted as $\Delta \boldsymbol{r} = \boldsymbol{0} - \boldsymbol{r}^k = -\boldsymbol{r}^k$. The matrix-vector product is calculated from data obtained during the previous quasi-Newton iterations. Line 12 shows that the flow equations and structural equations are solved in quasi-Newton iteration k, resulting in $\tilde{\boldsymbol{d}}^{k+1} = \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}(\boldsymbol{d}^{k+1})$ and the corresponding residual $\boldsymbol{r}^{k+1}$. The counter k is then increased on line 13. So, at the beginning of quasi-Newton iteration $k+1$, a set of known residual vectors

$$\boldsymbol{r}^k, \quad \boldsymbol{r}^{k-1}, \quad \ldots, \quad \boldsymbol{r}^1, \quad \boldsymbol{r}^0 \tag{18a}$$

Algorithm 1 The interface quasi-Newton algorithm with an approximation for the inverse of the Jacobian from a least-squares model (IQN-ILS) [27].

$k = 0$
$\boldsymbol{r}^0 = \tilde{\boldsymbol{d}}^0 - \boldsymbol{d}^0 = \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}(\boldsymbol{d}^0) - \boldsymbol{d}^0$
while $||\boldsymbol{r}^k||_2 > \varepsilon_o$ **do**
    if $k = 0$ and ($q = 0$ or $n = 0$) **then**
        $\boldsymbol{d}^{k+1} = \boldsymbol{d}^k + \omega \boldsymbol{r}^k$
    else
        construct $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ as in Eqs. (19), Eqs. (21) and Eqs. (22)
        calculate QR-decomposition $\boldsymbol{V}^k = \boldsymbol{Q}^k \boldsymbol{R}^k$
        solve $\boldsymbol{R}^k \boldsymbol{c}^k = -{\boldsymbol{Q}^k}^{\mathrm{T}} \boldsymbol{r}^k$
        $\boldsymbol{d}^{k+1} = \boldsymbol{d}^k + \boldsymbol{W}^k \boldsymbol{c}^k + \boldsymbol{r}^k$
    end if
    $\boldsymbol{r}^{k+1} = \tilde{\boldsymbol{d}}^{k+1} - \boldsymbol{d}^{k+1} = \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}(\boldsymbol{d}^{k+1}) - \boldsymbol{d}^{k+1}$
    $k = k + 1$
end while

and the corresponding set of vectors $\tilde{\boldsymbol{d}}$

$$\tilde{\boldsymbol{d}}^k, \quad \tilde{\boldsymbol{d}}^{k-1}, \quad \ldots, \quad \tilde{\boldsymbol{d}}^1, \quad \tilde{\boldsymbol{d}}^0 \tag{18b}$$

are available. The differences between the vectors $\boldsymbol{r}$ and $\tilde{\boldsymbol{d}}$ from the last coupling iteration and those from the first coupling iteration is calculated

$$\Delta \boldsymbol{r}^{k-1} = \boldsymbol{r}^k - \boldsymbol{r}^0 \tag{19a}$$

$$\Delta \tilde{\boldsymbol{d}}^{k-1} = \tilde{\boldsymbol{d}}^k - \tilde{\boldsymbol{d}}^0. \tag{19b}$$

This yields a set of differences $\Delta \boldsymbol{r}^i$

$$\Delta \boldsymbol{r}^{k-1}, \quad \Delta \boldsymbol{r}^{k-2}, \quad \ldots, \quad \Delta \boldsymbol{r}^1, \quad \Delta \boldsymbol{r}^0 \tag{20a}$$

as well as the corresponding set of differences $\Delta \tilde{\boldsymbol{d}}^i$

$$\Delta \tilde{\boldsymbol{d}}^{k-1}, \quad \Delta \tilde{\boldsymbol{d}}^{k-2}, \quad \ldots, \quad \Delta \tilde{\boldsymbol{d}}^1, \quad \Delta \tilde{\boldsymbol{d}}^0, \tag{20b}$$

which both grow with one vector in each coupling iteration.

The vectors $\Delta \boldsymbol{r}^i$ and the corresponding vectors $\Delta \tilde{\boldsymbol{d}}^i$ are stored as the columns of the matrices

$$\boldsymbol{A}^k = \begin{bmatrix} \Delta \boldsymbol{r}^{k-1} & \Delta \boldsymbol{r}^{k-2} & \ldots & \Delta \boldsymbol{r}^1 & \Delta \boldsymbol{r}^0 \end{bmatrix} \tag{21a}$$

and

$$\boldsymbol{B}^k = \begin{bmatrix} \Delta\tilde{\boldsymbol{d}}^{k-1} & \Delta\tilde{\boldsymbol{d}}^{k-2} & \dots & \Delta\tilde{\boldsymbol{d}}^{1} & \Delta\tilde{\boldsymbol{d}}^{0} \end{bmatrix}, \tag{21b}$$

respectively. If the behaviour of the problem is sufficiently similar in consecutive time steps, the data from the previous time steps can be reused. The matrices $\boldsymbol{A}^k$ and $\boldsymbol{B}^k$ are then combined with those from q previous time steps (if at least q time steps have already been performed), giving

$$\boldsymbol{V}^k = \begin{bmatrix} \boldsymbol{A}^k & \boldsymbol{A}^n & \dots & \boldsymbol{A}^{n-q+2} & \boldsymbol{A}^{n-q+1} \end{bmatrix} \tag{22a}$$

and

$$\boldsymbol{W}^k = \begin{bmatrix} \boldsymbol{B}^k & \boldsymbol{B}^n & \dots & \boldsymbol{B}^{n-q+2} & \boldsymbol{B}^{n-q+1} \end{bmatrix}. \tag{22b}$$

The number of columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ is indicated with b which is generally much smaller than the number of rows a. However, the data from previous time steps are only approximately correct for the current time step, even though the reuse of data from previous time steps results in faster convergence of the coupling iterations in several numerical experiments [30]. Cases with large differences between the time steps, for example, do not benefit from this reuse. To select the best value for q, the authors generally simulate the first 10 or 15 time steps with different values of the reuse parameter q, ranging from 0 to 10. The value q that results in the lowest number of coupling iterations is then used for the complete simulation.

The vector $\Delta\boldsymbol{r} = \boldsymbol{0} - \boldsymbol{r}^k$ is approximated as a linear combination of the known $\Delta\boldsymbol{r}^i$

$$\Delta\boldsymbol{r} \approx \boldsymbol{V}^k \boldsymbol{c}^k \tag{23}$$

with $\boldsymbol{c}^k \in \mathbb{R}^{b\times 1}$ the coefficients of the decomposition. Because $b \leq a$, Eq. (23) is an overdetermined set of equations for the elements of $\boldsymbol{c}^k$ and hence the least-squares solution to this linear system is calculated. In simulations with a low number of degrees of freedom on the interface, the number of columns in $\boldsymbol{V}^k$ has to be limited to a by discarding the rightmost columns. To solve the least-squares problem, the so-called economy-size QR-decomposition of $\boldsymbol{V}^k$ is calculated using Householder transformations [33]

$$\boldsymbol{V}^k = \boldsymbol{Q}^k \boldsymbol{R}^k, \tag{24}$$

with $\boldsymbol{Q}^k \in \mathbb{R}^{a\times b}$ an orthogonal matrix and $\boldsymbol{R}^k \in \mathbb{R}^{b\times b}$ an upper triangular matrix.

The coefficient vector $\boldsymbol{c}^k$ is then determined by solving the triangular system

$$\boldsymbol{R}^k \boldsymbol{c}^k = {\boldsymbol{Q}^k}^{\mathrm{T}} \Delta\boldsymbol{r} \tag{25}$$

using back substitution. If a $\Delta \boldsymbol{r}^i$ vector is (almost) a linear combination of other $\Delta \boldsymbol{r}^j$ vectors, one of the diagonal elements of $\boldsymbol{R}^k$ will (almost) be zero. Therefore, the equation corresponding to that row of $\boldsymbol{R}^k$ cannot be solved during the back substitution. If a small diagonal element is detected, the corresponding columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ are removed. Subsequently, the QR-decomposition (Eq. (24)) is performed again. This procedure is repeated until none of the diagonal elements is too small.

The tolerance ε_r for the detection of small diagonal elements depends on how accurately the flow equations and structural equations are solved. An appropriate value for ε_r can be determined by analyzing the change of the vector $\tilde{\boldsymbol{d}}$ due to a small perturbation of the vector $\boldsymbol{d}$. If the perturbation is too small, the resulting change will be numerical noise. The value of ε_r should be chosen so that the change of $\tilde{\boldsymbol{d}}$ has a physical meaning (i.e. it is no numerical noise) if the perturbation of $\boldsymbol{d}$ has an L^2-norm larger than ε_r. If the solution of the flow equations and the structural equations is calculated with more significant digits, for example by using stricter convergence criteria inside the solvers, then a smaller value of ε_r can be used. Especially for cases with small changes of the interface variables during the coupling iterations, it is important to set ε_r using the procedure described above.

The $\Delta \tilde{\boldsymbol{d}}$ that corresponds to $\Delta \boldsymbol{r}$ is subsequently calculated as a linear combination of the previous $\Delta \tilde{\boldsymbol{d}}^i$, analogous to Eq. (23), giving

$$\Delta \tilde{\boldsymbol{d}} = \boldsymbol{W}^k \boldsymbol{c}^k. \tag{26}$$

From Eq. (16), it follows that

$$\Delta \boldsymbol{r} = \Delta \tilde{\boldsymbol{d}} - \Delta \boldsymbol{d} \tag{27}$$

and substitution of Eq. (26) in Eq. (27) results in

$$\Delta \boldsymbol{d} = \boldsymbol{W}^k \boldsymbol{c}^k - \Delta \boldsymbol{r}. \tag{28}$$

Because the coefficients $\boldsymbol{c}^k$ are a function of $\Delta \boldsymbol{r}$, Eq. (28) shows how $\Delta \boldsymbol{d}$ can be approximated for a given $\Delta \boldsymbol{r}$. Hence, Eq. (28) can be seen as a procedure to calculate the product of the approximation for the inverse of the Jacobian and a vector $\Delta \boldsymbol{r} = -\boldsymbol{r}^k$

$$\Delta \boldsymbol{d} = \widehat{(\boldsymbol{\mathcal{R}}'^k)^{-1}} \Delta \boldsymbol{r} = \boldsymbol{W}^k \boldsymbol{c}^k + \boldsymbol{r}^k. \tag{29}$$

It can be proven that the procedure above corresponds to Newton iterations for the part of $\Delta \boldsymbol{r}$ in the span of the columns of $\boldsymbol{V}^k$ while Gauss-Seidel iterations are

performed for the part of $\Delta \boldsymbol{r}$ orthogonal to the span of the columns of $\boldsymbol{V}^k$. For the flow in a straight, flexible tube, these quasi-Newton iterations with a low-rank least-squares approximation for the inverse of the Jacobian converge quickly as only a fraction of the Fourier modes in the error on the fluid-structure interface is unstable during Gauss-Seidel iterations [34, 35].

Because the matrices $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ have to contain at least one column to perform the above computations, a relaxation with a fixed factor ω (line 5) is performed in the second coupling iteration of the first time step if data from the previous time steps is reused ($q > 0$) and in the second coupling iteration of each time step without reuse ($q = 0$). The value of ω can vary over a wide range with only a small influence on the number of coupling iterations. The lower bound for ω is determined by the finite accuracy and the solution tolerance of the solvers while the upper bound has to avoid divergence of the solvers or grid distortion.

3.2. IBQN-LS

The IBQN-LS coupling algorithm solves the fluid-structure interaction problem (Eq. (11)) written as

$$\begin{cases} \boldsymbol{\mathcal{F}}(\boldsymbol{d}) - \boldsymbol{s} = \boldsymbol{0} \\ \boldsymbol{\mathcal{S}}(\boldsymbol{s}) - \boldsymbol{d} = \boldsymbol{0} \end{cases} \tag{30}$$

with block Newton-Raphson iterations of the Gauss-Seidel type. The linearized system

$$\begin{bmatrix} \widehat{\boldsymbol{\mathcal{F}}'} & -\boldsymbol{I} \\ -\boldsymbol{I} & \widehat{\boldsymbol{\mathcal{S}}'} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{d} \\ \Delta \boldsymbol{s} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{\mathcal{F}}(\boldsymbol{d}) - \boldsymbol{s} \\ \boldsymbol{\mathcal{S}}(\boldsymbol{s}) - \boldsymbol{d} \end{bmatrix} \tag{31}$$

is thus first solved for $\Delta \boldsymbol{d}$, followed by an update of $\boldsymbol{d}$, the right-hand side and the approximate Jacobian of the flow solver. Subsequently, the modified system is solved for $\Delta \boldsymbol{s}$ and afterwards $\boldsymbol{s}$ is updated. As a consequence, the IBQN-LS method modifies the stress distribution that is calculated by the flow solver before transferring it to the structural solver. In agreement with the notation for intermediate values, the input and output of the flow solver are denoted as $\boldsymbol{d}^{k+1}$ and $\tilde{\boldsymbol{s}}^{k+1}$ and the input and output of the structural solver as $\boldsymbol{s}^{k+1}$ and $\tilde{\boldsymbol{d}}^{k+1}$. The complete IBQN-LS coupling algorithm is shown in Algorithm 2.

Starting from the displacement $\boldsymbol{d}^k$ that was given as input to the flow solver in the previous coupling iteration, the displacement $\boldsymbol{d}^{k+1} = \boldsymbol{d}^k + \Delta \boldsymbol{d}^k$ is calculated by solving the system

$$\left(\boldsymbol{I} - \widehat{\boldsymbol{\mathcal{S}}'^k} \widehat{\boldsymbol{\mathcal{F}}'^k} \right) \Delta \boldsymbol{d}^k = \tilde{\boldsymbol{d}}^k - \boldsymbol{d}^k + \widehat{\boldsymbol{\mathcal{S}}'^k} (\tilde{\boldsymbol{s}}^k - \boldsymbol{s}^k) \tag{32}$$

Algorithm 2 The interface block quasi-Newton algorithm with approximate Jacobians from least-squares models (IBQN-LS) [28].

$k = 0$
$\tilde{\boldsymbol{s}}^0 = \boldsymbol{\mathcal{F}}(\boldsymbol{d}^0)$
$\boldsymbol{s}^0 = \tilde{\boldsymbol{s}}^0$
$\tilde{\boldsymbol{d}}^0 = \boldsymbol{\mathcal{S}}(\boldsymbol{s}^0)$
$\boldsymbol{r}^0 = \tilde{\boldsymbol{d}}^0 - \boldsymbol{d}^0$
while $||\boldsymbol{r}^k||_2 > \varepsilon_o$ **do**
    if $k = 0$ and ($q = 0$ or $n = 0$) **then**
        $\boldsymbol{d}^{k+1} = \boldsymbol{d}^k + \omega \boldsymbol{r}^k$
    else
        construct $\boldsymbol{V}_s^k$ and $\boldsymbol{W}_s^k$
        calculate QR-decomposition $\boldsymbol{V}_s^k = \boldsymbol{Q}_s^k \boldsymbol{R}_s^k$
        solve Eq. (32) for $\Delta \boldsymbol{d}^k$
        $\boldsymbol{d}^{k+1} = \boldsymbol{d}^k + \Delta \boldsymbol{d}^k$
    end if
    $\tilde{\boldsymbol{s}}^{k+1} = \boldsymbol{\mathcal{F}}(\boldsymbol{d}^{k+1})$
    if $k = 0$ and ($q = 0$ or $n = 0$) **then**
        $\boldsymbol{s}^{k+1} = \tilde{\boldsymbol{s}}^{k+1}$
    else
        construct $\boldsymbol{V}_f^{k+1}$ and $\boldsymbol{W}_f^{k+1}$
        calculate QR-decomposition $\boldsymbol{V}_f^{k+1} = \boldsymbol{Q}_f^{k+1} \boldsymbol{R}_f^{k+1}$
        solve Eq. (37) for $\Delta \boldsymbol{s}^k$
        $\boldsymbol{s}^{k+1} = \boldsymbol{s}^k + \Delta \boldsymbol{s}^k$
    end if
    $\boldsymbol{r}^{k+1} = \tilde{\boldsymbol{d}}^{k+1} - \boldsymbol{d}^{k+1} = \boldsymbol{\mathcal{S}}(\boldsymbol{s}^{k+1}) - \boldsymbol{d}^{k+1}$
    $k = k + 1$
end while

for $\Delta \boldsymbol{d}^k$. This linear system is solved in a matrix-free way with an iterative Krylov solver like the generalized minimal residual (GMRES) method [36]. Consequently, the matrix on the left-hand side of Eq. (32) and thus the approximate Jacobians $\widehat{\boldsymbol{\mathcal{F}}'^k}$ and $\widehat{\boldsymbol{\mathcal{S}}'^k}$ do not have to be calculated explicitly; a procedure to calculate the product of these matrices with a vector is sufficient. This procedure is similar to the procedure described in Section 3.1. The matrix-vector product with $\widehat{\boldsymbol{\mathcal{F}}'^k}$ at the beginning of iteration $k+1$ is calculated from the previous inputs

$$\boldsymbol{d}^k, \quad \boldsymbol{d}^{k-1}, \quad \ldots, \quad \boldsymbol{d}^1, \quad \boldsymbol{d}^0 \tag{33a}$$

and the corresponding outputs

$$\tilde{\boldsymbol{s}}^k = \boldsymbol{\mathcal{F}}(\boldsymbol{d}^k), \quad \tilde{\boldsymbol{s}}^{k-1} = \boldsymbol{\mathcal{F}}(\boldsymbol{d}^{k-1}), \quad \ldots, \quad \tilde{\boldsymbol{s}}^1 = \boldsymbol{\mathcal{F}}(\boldsymbol{d}^1), \quad \tilde{\boldsymbol{s}}^0 = \boldsymbol{\mathcal{F}}(\boldsymbol{d}^0) \tag{33b}$$

of the flow solver. The difference between the vectors $\boldsymbol{d}$ and $\tilde{\boldsymbol{s}}$ from the last coupling iteration and those from the first coupling iteration is calculated

$$\Delta \boldsymbol{d}^{k-1} = \boldsymbol{d}^k - \boldsymbol{d}^0 \tag{34a}$$

$$\Delta \tilde{\boldsymbol{s}}^{k-1} = \tilde{\boldsymbol{s}}^k - \tilde{\boldsymbol{s}}^0. \tag{34b}$$

All $\Delta \boldsymbol{d}^i$ and $\Delta \tilde{\boldsymbol{s}}^i$ ($i = 0, \ldots, k-1$) from the current time step (and possibly from previous time steps) are stored as columns of the matrices $\boldsymbol{V}_f^k$ and $\boldsymbol{W}_f^k$, with the subscript f referring to the flow solver. Subsequently, the economy-size QR-decomposition of $\boldsymbol{V}_f^k$ is calculated. To determine the product of $\widehat{\boldsymbol{\mathcal{F}}'^k}$ with a vector $\Delta \boldsymbol{d}$, the triangular system

$$\boldsymbol{R}_f^k \boldsymbol{c}_f^k = {\boldsymbol{Q}_f^k}^{\mathrm{T}} \Delta \boldsymbol{d} \tag{35}$$

is solved for $\boldsymbol{c}_f^k$, after which the matrix-vector product is calculated as

$$\widehat{\boldsymbol{\mathcal{F}}'^k} \Delta \boldsymbol{d} = \boldsymbol{W}_f^k \boldsymbol{c}_f^k. \tag{36}$$

The product of $\widehat{\boldsymbol{\mathcal{S}}'^k}$ with a vector is calculated analogously, based on the inputs and outputs of the structural solver.

Once $\boldsymbol{d}^{k+1}$ has been obtained, the corresponding stress distribution $\tilde{\boldsymbol{s}}^{k+1} = \boldsymbol{\mathcal{F}}(\boldsymbol{d}^{k+1})$ is calculated and the matrices $\boldsymbol{V}_f^{k+1}$, $\boldsymbol{W}_f^{k+1}$, $\boldsymbol{Q}_f^{k+1}$ and $\boldsymbol{R}_f^{k+1}$ are constructed. To calculate the stress distribution $\boldsymbol{s}^{k+1} = \boldsymbol{s}^k + \Delta \boldsymbol{s}^k$ that has to be applied on the structure, the system

$$\left(\boldsymbol{I} - \widehat{\boldsymbol{\mathcal{F}}'^{k+1}} \widehat{\boldsymbol{\mathcal{S}}'^k} \right) \Delta \boldsymbol{s}^k = \tilde{\boldsymbol{s}}^{k+1} - \boldsymbol{s}^k + \widehat{\boldsymbol{\mathcal{F}}'^{k+1}} (\tilde{\boldsymbol{d}}^k - \boldsymbol{d}^{k+1}) \tag{37}$$

is solved, again with the matrix-free iterative solver. Each time the solution to either the flow problem or the structural problem has been calculated, the procedure for the product of the corresponding solver's approximate Jacobian with a vector is improved by means of that solver's latest input and output.

Analogous to the IQN-ILS technique, the matrices $\boldsymbol{V}_{f,s}^k$ and $\boldsymbol{W}_{f,s}^k$ have to contain at least one column to calculate the quasi-Newton update; otherwise a relaxation with a fixed factor ω is used for the interface's displacement (line 8 in Algorithm 2) and the stress distribution is passed on without modification (line 17).

4. Multi-solver quasi-Newton coupling algorithms

In the IQN-ILS and IBQN-LS algorithms, the columns of a matrix $\boldsymbol{V}$ are differences of an input while the columns of the corresponding matrix $\boldsymbol{W}$ are the differences of the corresponding output. The relation between the columns of $\boldsymbol{A}^n$ and $\boldsymbol{B}^n$ is only approximate at t^{n+1}. Nevertheless, the columns of $\boldsymbol{A}^n$ can be used during the coupling iterations at t^{n+1}. In most cases, it is not feasible to perturb each degree-of-freedom on the fluid-structure interface consecutively to obtain a finite difference approximation for the Jacobian. However, the columns of the matrix $\boldsymbol{A}^n$ contain specific combinations of the degrees-of-freedom on the interface that accelerated the convergence of the coupling iterations in the previous time step. Hence, it is expected that knowing the difference of the output at t^{n+1} due to the same difference of the input as used at time level t^n will improve the least-squares model for the approximate Jacobian.

The difference of output at t^{n+1} due to a difference of input from t^n can be calculated exactly by applying this difference again at t^{n+1}. Moreover, the recalculation of differences from previous time steps can be done in parallel with normal coupling iterations if $g > 1$ flow solvers and $h > 1$ structural solvers are used. In the following sections, a subscript i or j distinguishes the different solvers and their respective input and output.

4.1. MS-IQN-ILS

Algorithm 3 describes the Multi-Solver IQN-ILS (MS-IQN-ILS) algorithm with parallel recalculation of differences from the previous time step. Solvers $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ calculate the solution of the coupled problem, while solvers $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) recalculate differences from previous time steps.

Lines 5 to 14 describe the standard IQN-ILS algorithm, with the exception of the 'start' that has been added on line 13. This command means that the calculation has to be started, without waiting for the result to continue the execution

of the coupling algorithm. On line 4, the coupling algorithm checks whether $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ have completed the previous calculation, i.e. whether they are 'ready', before starting the following calculation. The check of the convergence tolerance on line 3 evaluates to false while the calculation of $\boldsymbol{r}_1^k$ is ongoing.

Because the coupling code is not waiting on line 13 until $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ are ready, it can control the other solvers in the meantime. On line 16, the coupling algorithm loops over the additional solvers $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and on line 23, it loops over the additional solvers $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$). In a first step of the recalculation of differences by solver $\boldsymbol{\mathcal{F}}_i$ and $\boldsymbol{\mathcal{S}}_j$, a column $\Delta\boldsymbol{r}$ of $\boldsymbol{A}^n$ and the corresponding column $\Delta\tilde{\boldsymbol{d}}$ of $\boldsymbol{B}^n$ are selected. In this case, the newest columns are selected first. As a result, the leftmost columns of $\boldsymbol{A}^n$ and $\boldsymbol{B}^n$ are selected first for $\boldsymbol{\mathcal{F}}_2$, followed by the second leftmost columns for $\boldsymbol{\mathcal{F}}_3$, etc. Other selection procedures are also possible: oldest first, largest $||\Delta\boldsymbol{r}||_2$ first, largest $||\Delta\tilde{\boldsymbol{d}}||_2/||\Delta\boldsymbol{r}||_2$ first, etc.

Subsequently, the selected $\Delta\boldsymbol{r}$ and $\Delta\tilde{\boldsymbol{d}}$ from t^n have to be recalculated at t^{n+1}. These vectors are differences with respect to the reference vectors $\boldsymbol{r}_1^{n,0}$ and $\tilde{\boldsymbol{d}}_1^{n,0}$ which originate from the first coupling iteration at t^n. To be recalculated at t^{n+1}, the vectors $\Delta\boldsymbol{r}$ and $\Delta\tilde{\boldsymbol{d}}$ have to be added to a value of $\boldsymbol{r}$ and $\tilde{\boldsymbol{d}}$ at t^{n+1}, namely $\boldsymbol{r}_i^0$ and $\tilde{\boldsymbol{d}}_i^0$. Here, extrapolated values $\boldsymbol{r}_i^0 = \boldsymbol{0}$ and $\tilde{\boldsymbol{d}}_i^0 = \boldsymbol{d}_1^0$ (as calculated in Eq. (12)) are used. As a result, the recalculation of differences from the previous time step can begin immediately at the start of the new time step, simultaneously with the first coupling iteration between $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$. If the result of the first calculation of $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$, i.e. $\boldsymbol{r}_1^0$ and $\tilde{\boldsymbol{d}}_1^0$, were used as $\boldsymbol{r}_i^0$ and $\tilde{\boldsymbol{d}}_i^0$, the recalculation would have to wait until these values have been calculated.

So, the difference $\Delta\boldsymbol{r}$ is added to the extrapolated value of $\boldsymbol{r}$ in the current time step, being $\boldsymbol{r}_i^0 = \boldsymbol{0}$.

$$\boldsymbol{r}_i = \boldsymbol{r}_i^0 + \Delta\boldsymbol{r} \tag{38}$$

Similarly, $\Delta\tilde{\boldsymbol{d}}$ is added to $\tilde{\boldsymbol{d}}_i^0$, the extrapolated interface displacement.

$$\tilde{\boldsymbol{d}}_i = \tilde{\boldsymbol{d}}_i^0 + \Delta\tilde{\boldsymbol{d}} \tag{39}$$

Using $\boldsymbol{r} = \tilde{\boldsymbol{d}} - \boldsymbol{d}$, the input for $\boldsymbol{\mathcal{F}}_i$ is then calculated as

$$\boldsymbol{d}_i = \tilde{\boldsymbol{d}}_i - \boldsymbol{r}_i. \tag{40}$$

The calculation of solver $\boldsymbol{\mathcal{F}}_i$ with the $\boldsymbol{d}_i$ from Eq. (40) as input is started on line 20. Again, the coupling algorithm does not wait until this calculation is complete. Instead, it checks on line 17 whether $\boldsymbol{\mathcal{F}}_i$ has completed its calculation. When this is the case, $\boldsymbol{d}_i$ and the corresponding $\tilde{\boldsymbol{s}}_i$ are added to an intermediate first in, first out (FIFO) queue.

Algorithm 3 The multi-solver IQN-ILS (MS-IQN-ILS) algorithm.

$k = 0$
start $\boldsymbol{r}_1^0 = \tilde{\boldsymbol{d}}_1^0 - \boldsymbol{d}_1^0 = \boldsymbol{\mathcal{S}}_1 \circ \boldsymbol{\mathcal{F}}_1(\boldsymbol{d}_1^0) - \boldsymbol{d}_1^0$
while $||\boldsymbol{r}_1^k||_2 > \varepsilon_o$ **do**
    if $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ are ready **then**
        if $k = 0$ and ($q = 0$ or $n = 0$) **then**
            $\boldsymbol{d}_1^{k+1} = \boldsymbol{d}_1^k + \omega \boldsymbol{r}_1^k$
        else
            construct $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$
            calculate QR-decomposition $\boldsymbol{V}^k = \boldsymbol{Q}^k \boldsymbol{R}^k$
            solve $\boldsymbol{R}^k \boldsymbol{c}^k = -{\boldsymbol{Q}^k}^{\mathrm{T}} \boldsymbol{r}_1^k$
            $\boldsymbol{d}_1^{k+1} = \boldsymbol{d}_1^k + \boldsymbol{W}^k \boldsymbol{c}^k + \boldsymbol{r}_1^k$
        end if
        start $\boldsymbol{r}_1^{k+1} = \tilde{\boldsymbol{d}}_1^{k+1} - \boldsymbol{d}_1^{k+1} = \boldsymbol{\mathcal{S}}_1 \circ \boldsymbol{\mathcal{F}}_1(\boldsymbol{d}_1^{k+1}) - \boldsymbol{d}_1^{k+1}$
        $k = k + 1$
    end if
    for $i = 2$ to g **do**
        if $\boldsymbol{\mathcal{F}}_i$ is ready **then**
            select $\Delta \boldsymbol{r}$ and $\Delta \tilde{\boldsymbol{d}}$
            $\boldsymbol{d}_i = (\boldsymbol{d}_i^0 + \Delta \tilde{\boldsymbol{d}}) - (\boldsymbol{r}_i^0 + \Delta \boldsymbol{r})$
            start $\tilde{\boldsymbol{s}}_i = \boldsymbol{\mathcal{F}}_i(\boldsymbol{d}_i)$
        end if
    end for
    for $j = 2$ to h **do**
        if $\boldsymbol{\mathcal{S}}_j$ is ready **then**
            get $\boldsymbol{d}_j$ and $\tilde{\boldsymbol{s}}_j$ from the intermediate FIFO queue
            start $\boldsymbol{r}_j = \tilde{\boldsymbol{d}}_j - \boldsymbol{d}_j = \boldsymbol{\mathcal{S}}_j(\tilde{\boldsymbol{s}}_j) - \boldsymbol{d}_j$
        end if
    end for
end while
for $i = 2$ to g **do**
    start synchronizing $\boldsymbol{\mathcal{F}}_i$ with $\boldsymbol{\mathcal{F}}_1$
end for
for $j = 2$ to h **do**
    start synchronizing $\boldsymbol{\mathcal{S}}_j$ with $\boldsymbol{\mathcal{S}}_1$
end for

If a structural solver $\boldsymbol{\mathcal{S}}_j$ is ready (line 24), it takes the oldest $\boldsymbol{d}_j$ and the corresponding $\tilde{\boldsymbol{s}}_j$ from the intermediate FIFO queue. The structural solver then starts to calculate the interface displacement $\tilde{\boldsymbol{d}}_j = \boldsymbol{\mathcal{S}}_j(\tilde{\boldsymbol{s}}_j)$ and the corresponding residual $\boldsymbol{r}_j = \tilde{\boldsymbol{d}}_j - \boldsymbol{d}_j$. This intermediate FIFO queue thus decouples the flow solvers $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) from the structural solvers $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) and enables a different number of flow solvers and structural solvers ($g \neq h$). If, for example, a flow calculation takes significantly longer than a structural calculation, then more flow solvers than structural solvers can be used as the additional flow solvers and additional structural solvers have to recalculate the same number of modes.

When the calculation of $\boldsymbol{\mathcal{S}}_j$ has completed, both the residual $\boldsymbol{r}_j$ and the output $\tilde{\boldsymbol{d}}_j$ are known. By subtracting the references for differences at t^{n+1}, respectively $\boldsymbol{r}_1^0$ and $\tilde{\boldsymbol{d}}_1^0$, a new mode in the current time step becomes available.

$$\Delta\boldsymbol{r} = \boldsymbol{r}_j - \boldsymbol{r}_1^0 \tag{41a}$$

$$\Delta\tilde{\boldsymbol{d}} = \tilde{\boldsymbol{d}}_j - \tilde{\boldsymbol{d}}_1^0 \tag{41b}$$

The vectors $\boldsymbol{r}_1^0$ and $\tilde{\boldsymbol{d}}_1^0$ are the first residual and the first output of $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$. As long as these vectors are unknown because the first calculation of $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ at t^{n+1} is ongoing, the data calculated by $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and $\boldsymbol{\mathcal{S}}_j$ ($i = j, \ldots, h$) are stored in a second queue. Once the reference values have been calculated, the differences corresponding to the data in this second queue are calculated. These differences have to be calculated with respect to vectors $\boldsymbol{r}$ and $\tilde{\boldsymbol{d}}$ that have actually been calculated, such as the first residual and output of $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$. If these differences were calculated with respect to the extrapolation, they would not characterize the system because the extrapolation is only an approximation for the residual and output at t^{n+1} and not a calculated value.

All differences from the previous time step that have been recalculated are stored as columns of the matrices $\boldsymbol{C}^n$ and $\boldsymbol{D}^n$. These matrices are then combined with $\boldsymbol{A}^k$ and $\boldsymbol{B}^k$ to form $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ as

$$\boldsymbol{V}^k = \begin{bmatrix} \boldsymbol{A}^k & \boldsymbol{C}^n \end{bmatrix} \tag{42a}$$

and

$$\boldsymbol{W}^k = \begin{bmatrix} \boldsymbol{B}^k & \boldsymbol{D}^n \end{bmatrix}. \tag{42b}$$

The columns of $\boldsymbol{A}^k$ and $\boldsymbol{B}^k$ contain the differences calculated by $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ in the current time step. Differences from previous time steps that have not been recalculated (yet) can be included in the matrices $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ as well, but this is not done in this work. In that case, the old difference should be removed once it

has been recalculated so that the same difference is not present twice. Because no differences from previous time steps that have not been recalculated at t^{n+1} are included in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$, these matrices are empty at the beginning of each time step.

All differences calculated by $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ in the current time step and all differences that have been recalculated by $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) in the current time step are candidates to be recalculated in the following time step. Two mechanisms avoid an ever increasing number of differences. The first one is that once $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ have found the correct solution, the coupling discards old differences that have not been recalculated at t^{n+1}. However, before discarding these differences, the coupling waits until all solvers that are recalculating differences have completed their current calculation as it is practically difficult to stop the solvers during their calculation. It is of course possible to select a wider window for the differences that can be recalculated, for example differences that have been calculated in the last two or three time steps. The second mechanism is the tolerance ε_r for the detection of small diagonal elements in $\boldsymbol{R}^k$. If a small diagonal element is detected, the corresponding columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ are removed, which means that they cannot be recalculated in the following time step.

When the convergence criterion has been satisfied, the current implementation of the coupling algorithm waits until all calculations of the additional solvers are ready, as it is difficult to stop the solvers during a calculation. The differences that have been recalculated in these calculations are no longer required for the coupling iterations in the current time step, but they are nonetheless added to $\boldsymbol{C}^n$ and $\boldsymbol{D}^n$ so that the can be recalculated once more in the following time step.

A last important aspect of the MS-IQN-ILS algorithm is the synchronization of the solvers $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) with $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$, as mentioned on lines 30 to 35 of Algorithm 3. At the end of each time step, the values of the degrees of freedom inside the fluid and solid domain have to be the same in all solvers. Otherwise, unphysical results will be obtained in the following time step. For example, without synchronization, the stress distribution on the interface for a given displacement would depend on which flow solver is used ($\boldsymbol{\mathcal{F}}_i(\boldsymbol{d}) \neq \boldsymbol{\mathcal{F}}_j(\boldsymbol{d})$ if $i \neq j$) because the solution at $t^n, t^{n-1}, \ldots$ influences the solution at t^{n+1}. These difficulties can be avoided by copying the degrees of freedom in the entire fluid and solid domain from $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ to all other flow solvers and structural solvers, once the coupling iterations have converged. If the implementation does not allow to copy the degrees of freedom from one solver to another one or to read a file with all values from another solver, the same result can be obtained by solving the equations once more in $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and

$\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) with $\boldsymbol{d}_1^{last}$ and $\boldsymbol{s}_1^{last}$ as input. This implementation will of course reduce the gain of the parallel recalculation of differences as one additional calculation (equivalent with one coupling iteration) would have to be performed in each time step. The synchronization of the different solvers can of course be done in parallel. The coupling code does not have to wait until one solver has completed the synchronization to start synchronizing the following solver.

4.2. MS-IBQN-LS

Algorithm 4 describes the Multi-Solver IBQN-LS (MS-IBQN-LS) algorithm with parallel recalculation of differences from the previous time step. Also in this coupling algorithm, solvers $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ calculate the solution of the coupled problem, while solvers $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) recalculate differences from the previous time step.

Lines 8 to 16 and lines 19 to 27 describe the standard IBQN-LS algorithm, with the exception of the 'start' on line 16 and line 27. On line 5, the coupling algorithm checks whether $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ are ready, i.e. whether $\boldsymbol{\mathcal{F}}_1$ has completed its previous calculation and $\boldsymbol{\mathcal{S}}_1$ can begin the following calculation or vice versa. The variable ℓ alternates between 0 and 1 to ensure that $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ take turns. The check of the convergence tolerance on line 4 again evaluates to false while the calculation of $\boldsymbol{r}_1^k$ is ongoing.

To obtain the difference that has to be recalculated by solver $\boldsymbol{\mathcal{F}}_i$, a column $\Delta\boldsymbol{d}$ of $\boldsymbol{A}_f^n$ is selected in the same way as explained in the previous section. This difference is added to the vector $\boldsymbol{d}_i^0$, the extrapolated interface displacement (as calculated in Eq. (12)).

$$\boldsymbol{d}_i = \boldsymbol{d}_i^0 + \Delta\boldsymbol{d} \tag{43}$$

As a result, the recalculation of differences can start immediatialy at the beginning of the time step. The coupling code checks on line 32 whether $\boldsymbol{\mathcal{F}}_i$ has completed its calculation. When this is the case, both the displacement $\boldsymbol{d}_i$ and the corresponding stress distribution $\tilde{\boldsymbol{s}}_i$ are known. By subtracting the first input and output of $\boldsymbol{\mathcal{F}}_1$, respectively $\boldsymbol{d}_1^0$ and $\tilde{\boldsymbol{s}}_1^0$, a new difference of the flow solver in the current time step becomes available.

$$\Delta\boldsymbol{d} = \boldsymbol{d}_i - \boldsymbol{d}_1^0 \tag{44a}$$

$$\Delta\tilde{\boldsymbol{s}} = \tilde{\boldsymbol{s}}_i - \tilde{\boldsymbol{s}}_1^0 \tag{44b}$$

The data calculated by the additional flow solvers $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) are stored in a queue as long as the vectors $\boldsymbol{d}_1^0$ and $\tilde{\boldsymbol{s}}_1^0$ are unknown because the first calculation of $\boldsymbol{\mathcal{F}}_1$ is ongoing. The corresponding differences are calculated as soon as $\boldsymbol{d}_1^0$

Algorithm 4 The multi-solver IBQN-LS (MS-IBQN-LS) algorithm (Part 1).

$k = 0$
$\ell = 0$
$\boldsymbol{r}_1^0 = \tilde{\boldsymbol{d}}_1^1 - \boldsymbol{d}_1^0 = \boldsymbol{\mathcal{S}}_1 \circ \boldsymbol{\mathcal{F}}_1(\boldsymbol{d}_1^0) - \boldsymbol{d}_1^0$
while $||\boldsymbol{r}_1^k||_2 > \varepsilon_o$ **do**
    if $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ are ready **then**
        if $\ell = 0$ **then**
            $\ell = 1$
            if $k = 0$ and ($q = 0$ or $n = 0$) **then**
                $\boldsymbol{d}_1^{k+1} = \boldsymbol{d}_1^k + \omega \boldsymbol{r}_1^k$
            else
                construct $\boldsymbol{V}_s^k$ and $\boldsymbol{W}_s^k$
                calculate QR-decomposition $\boldsymbol{V}_s^k = \boldsymbol{Q}_s^k \boldsymbol{R}_s^k$
                solve Eq. (32) for $\Delta \boldsymbol{d}^k$
                $\boldsymbol{d}_1^{k+1} = \boldsymbol{d}_1^k + \Delta \boldsymbol{d}^k$
            end if
            start $\tilde{\boldsymbol{s}}_1^{k+1} = \boldsymbol{\mathcal{F}}_1(\boldsymbol{d}_1^{k+1})$
        else
            $\ell = 0$
            if $k = 0$ and ($q = 0$ or $n = 0$) **then**
                $\boldsymbol{s}_1^{k+1} = \tilde{\boldsymbol{s}}_1^{k+1}$
            else
                construct $\boldsymbol{V}_f^{k+1}$ and $\boldsymbol{W}_f^{k+1}$
                calculate QR-decomposition $\boldsymbol{V}_f^{k+1} = \boldsymbol{Q}_f^{k+1} \boldsymbol{R}_f^{k+1}$
                solve Eq. (37) for $\Delta \boldsymbol{s}^k$
                $\boldsymbol{s}_1^{k+1} = \boldsymbol{s}_1^k + \Delta \boldsymbol{s}^k$
            end if
            start $\boldsymbol{r}_1^{k+1} = \tilde{\boldsymbol{d}}_1^{k+1} - \boldsymbol{d}_1^{k+1} = \boldsymbol{\mathcal{S}}_1(\boldsymbol{s}_1^{k+1}) - \boldsymbol{d}_1^{k+1}$
            $k = k + 1$
        end if
    end if

Algorithm 4 The multi-solver IBQN-LS (MS-IBQN-LS) algorithm (Part 2).

```
    for i = 2 to g do
        if F_i is ready then
            select Δd
            d_i = d_i^0 + Δd
            start s̃_i = F_i(d_i)
        end if
    end for
    for j = 2 to h do
        if S_j is ready then
            select Δs
            s_j = s_j^0 + Δs
            start d̃_j = S_j(s_j)
        end if
    end for
end while
for i = 2 to g do
    start synchronizing F_i with F_1
end for
for j = 2 to h do
    start synchronizing S_j with S_j
end for
```

and $\tilde{\boldsymbol{s}}_1^0$ are known. Again, the differences have to be calculated with respect to an input and output that actually comes from a flow solver and not with respect to the extrapolation.

All differences from the previous time step that have been recalculated are stored as columns of the matrices $\boldsymbol{C}_f^n$ and $\boldsymbol{D}_f^n$. These matrices are then combined with $\boldsymbol{A}_f^k$ and $\boldsymbol{B}_f^k$, which contain the differences calculated by $\boldsymbol{\mathcal{F}}_1$, to form $\boldsymbol{V}_f^k$ and $\boldsymbol{W}_f^k$ as

$$\boldsymbol{V}_f^k = \begin{bmatrix} \boldsymbol{A}_f^k & \boldsymbol{C}_f^n \end{bmatrix} \tag{45a}$$

and

$$\boldsymbol{W}_f^k = \begin{bmatrix} \boldsymbol{B}_f^k & \boldsymbol{D}_f^n \end{bmatrix}. \tag{45b}$$

An analogous procedure is followed for the additional structural solvers. There-

fore, an extrapolation of the stress on the interface is used, namely

$$\boldsymbol{s}_i^0 = \frac{5}{2}\boldsymbol{s}^n - 2\boldsymbol{s}^{n-1} + \frac{1}{2}\boldsymbol{s}^{n-2}, \tag{46}$$

similar to Eq. (12). The data calculated by the additional structural solvers $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) are stored in a second queue as long as the vectors $\boldsymbol{s}_1^0$ and $\tilde{\boldsymbol{d}}_1^0$ are unknown because the first calculation of $\boldsymbol{\mathcal{S}}_1$ is ongoing. Once the vectors $\boldsymbol{s}_1^0$ and $\tilde{\boldsymbol{d}}_1^0$ are known, the corresponding differences are calculated and added to $\boldsymbol{V}_s^k$ and $\boldsymbol{W}_s^k$.

Like in the MS-IQN-ILS algorithm, it is important to synchronize the solvers $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) with $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$, as mentioned on lines 46 to 51. Also for this coupling algorithm, the synchronization of all flow solvers and structural solvers can be performed in parallel.

5. Numerical results

All numerical results have been obtained using dedicated cluster nodes with two quad-core Intel Xeon X5355 processors and 16GB of working memory.

5.1. Flow in a one-dimensional flexible tube

The first example is the unsteady, incompressible flow in a straight, flexible tube with a circular cross-section and length L, depicted in Figure 3. This example is straightforward to implement and yet it is a representative test for a coupling technique [34]. The numerical model is one-dimensional and gravity and viscosity are not taken into account. Eqs. (1) are reformulated in conservative form for a deforming control volume, giving

$$\frac{\partial a}{\partial t} + \frac{\partial av}{\partial z} = 0 \tag{47a}$$

$$\frac{\partial av}{\partial t} + \frac{\partial av^2}{\partial z} + \frac{1}{\rho_f}\left(\frac{\partial ap}{\partial z} - p\frac{\partial a}{\partial z}\right) = 0 \tag{47b}$$

with z the coordinate along the axis of the tube, $a = \pi r^2$ the cross-sectional area of the tube and r the inner radius. t is the time and v the velocity along the axis of the tube.

The behaviour of the elastic tube wall is described with a Hookean constitutive relation. The structure contains no mass, as the inertia of the tube wall is neglected with regard to that of the fluid. An axisymmetric model is used in the coordinate

system (r,φ,z), with φ the angle in the cross-sectional plane as indicated in Figure 3. The stress in the tube wall in circumferential direction $\sigma_{\varphi\varphi}$ is approximated as

$$\sigma_{\varphi\varphi} = E\frac{r - r_o}{r_o} + \sigma_{\varphi\varphi o} \tag{48}$$

with E the Young's modulus and r_o the radius for which $\sigma_{\varphi\varphi} = \sigma_{\varphi\varphi o}$. As other stress components are neglected, this model only allows for radial motion of the tube wall. The force balance on the fluid-structure interface is

$$rp = \sigma_{\varphi\varphi} h \tag{49}$$

with h the thickness of the tube wall. By substituting the constitutive equation (Eq. (48)), $r_o p_o = \sigma_{\varphi\varphi o} h$ and $a = \pi r^2$ in Eq. (49), the following relation holds

$$a = a_o \left(\frac{\frac{p_o}{2\rho_f} - c_{MK}^2}{\frac{p}{2\rho_f} - c_{MK}^2} \right)^2 \tag{50}$$

with the Moens-Korteweg wave speed given by

$$c_{MK} = \sqrt{\frac{Eh}{2\rho_f r_o}}. \tag{51}$$

The resulting wave speed c is

$$c^2 = \frac{a}{\rho_f \frac{\mathrm{d}a}{\mathrm{d}p}} = c_{MK}^2 - \frac{p}{2\rho_f}, \tag{52}$$

which is used to impose a non-reflecting boundary

$$\frac{\mathrm{d}v}{\mathrm{d}t} = \frac{1}{c\rho_f}\frac{\mathrm{d}p}{\mathrm{d}t} \tag{53}$$

at the outlet.

The tube is discretized using a one-dimensional grid with $N = 100$ cells of length Δz, as indicated in Figure 3. As the fluid and the structure are discretized in the same way, no interpolation on the fluid-structure interface is required. The fluid velocity and pressure are stored in the cell centres. Central discretization is used for all terms in the continuity and momentum equation, except for the convective term in the momentum equation which is discretized with a first-order upwind scheme. The time discretization scheme is backward Euler and the time

step is indicated with Δt. The conservation of mass and momentum in a control volume around cell centre i is expressed by the following system of equations

$$\frac{\Delta z}{\Delta t}\left(a_i - a_i^n\right) + v_{i+1/2}a_{i+1/2} - v_{i-1/2}a_{i-1/2} - \frac{\alpha}{\rho_f}\left(p_{i+1} - 2p_i + p_{i-1}\right) = 0 \quad \text{(54a)}$$

$$\frac{\Delta z}{\Delta t}\left(v_i a_i - v_i^n a_i^n\right) + v_i v_{i+1/2}a_{i+1/2} - v_{i-1}v_{i-1/2}a_{i-1/2} + \frac{1}{2\rho_f}\left(a_{i+1/2}\left(p_{i+1} - p_i\right) + a_{i-1/2}\left(p_i - p_{i-1}\right)\right) = 0 \quad \text{(54b)}$$

for $v_i \geq 0$. The subscripts i, $i+1$ and $i-1$ indicate the cell centres ($i = 1, \ldots, N$) and the subscript $i \pm 1/2$ signifies the values calculated at the cell interfaces, $v_{i-1/2} = (v_{i-1} + v_i)/2$ and $v_{i+1/2} = (v_i + v_{i+1})/2$. A pressure stabilization term [37] with coefficient $\alpha = a_o/\left(v_o + \Delta z/\Delta t\right)$ has been added in the continuity equation to prohibit pressure wiggles due to central discretization of the pressure in the momentum equation, with v_o the reference flow velocity.

The equation for the structure (Eq. (50)) does not require further discretization as it can directly be used to calculate the cross-sectional area of a segment for a given pressure in that segment. The pressure-outlet condition in Eq. (53) is discretized as

$$p_{out} = 2\rho_f\left[c_{MK}^2 - \left(\sqrt{c_{MK}^2 - \frac{p_{out}^n}{2\rho_f}} - \frac{v_{out} - v_{out}^n}{4}\right)^2\right], \quad (55)$$

which takes into account the dependence of c on p (Eq. (52)) in the integration from time level n to $n+1$. At the inlet, the velocity is imposed as

$$v_{in} = v_o + \frac{v_o}{10}\sin^2(\pi n\tau) \quad (56)$$

with τ the dimensionless time step (defined in Eq. (58)). The pressure at the inlet and the velocity at the outlet are linearly extrapolated

$$p_{in} = 2p_1 - p_2 \quad \text{(57a)}$$

$$v_{out} = 2v_N - v_{N-1}. \quad \text{(57b)}$$

The parameters of this case are grouped in two dimensionless numbers, namely

$$\kappa = \frac{c_o}{v_o} = \frac{\sqrt{\frac{Eh}{2\rho_f r_o} - \frac{p_o}{2\rho_f}}}{v_o} \quad \text{and} \quad \tau = \frac{v_o \Delta t}{L}. \quad (58)$$

In all simulations presented in this section, the dimensionless stiffness is $\kappa = 10$ and the dimensionless time step is $\tau = 0.0025$. One period of the inlet boundary condition, i.e. 400 time steps, is simulated. The initial conditions are a dimensionless velocity of $v/c_o = v_o/c_o = 0.1$ (compatible with the inlet boundary condition), a dimensionless cross-sectional area of $a/a_o = 1$ and a dimensionless pressure of $p/(\rho_f c_o^2) = 0$.

As opposed to existing coupling algorithms like IQN-ILS, IBQN-LS, Aitken relaxation and Interface-GMRES, the multi-solver algorithms will need a slightly different number of coupling iterations each time the same simulation is performed. The parallel recalculation involves 'start' and 'ready' commands, so the order of the various calculations can change. Depending on whether a recalculated difference becomes available before or after $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ start a new calculation, the convergence will be slightly faster or slower. Therefore, all simulations have been performed 25 times. The number of coupling iterations per time step has been averaged over all time steps in a simulation with a given number of solvers and over all 25 runs of that simulation. The difference between the average number of coupling iterations per time step in two different runs with the same number of solvers was never more than one iteration per time step. Compared to the reduction of the average number of coupling iterations per time step as the number of solvers increases, the difference between the 25 runs of a simulation with the same number of solvers is thus small.

The speed-up of a simulation as a result of the multi-solver approach can be observed in the average number of coupling iterations between $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ per time step. The additional solvers $\boldsymbol{\mathcal{F}}_i$ ($i = 2, \ldots, g$) and $\boldsymbol{\mathcal{S}}_j$ ($j = 2, \ldots, h$) help $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ to find the solution of the coupled problem more quickly. Figure 4 depicts the average number of coupling iterations between $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ per time step for the MS-IQN-ILS and MS-IBQN-LS algorithm. The convergence criterion for the coupling iterations is $||\boldsymbol{r}_1^k||_2 < 10^{-3}||\boldsymbol{r}_1^0||_2$. As the number of solvers increases from $g = h = 1$ to $g = h = 8$, the average number of coupling iterations per time step decreases from 8 to 3 for MS-IQN-ILS and from 7 to 3 or 4 for MS-IBQN-LS. With a negligible duration of the communication and synchronization compared to the duration of the calculation, this will result in a reduction of the run time by at least 50 %. Not more than eight solvers of each type have been used because the curves flatten out as the number of solvers increases.

Figure 5 shows that the average number of coupling iterations per time step is almost constant for ω in the range 10^{-6} to 10^{-2}. In all other simulations for this first test case, ω is set to 10^{-2}. The influence of the tolerance ϵ_r for the detection of small diagonal elements is depicted in Figure 6 for MS-IQN-ILS. As

can be observed from this figure, the sensitivity of the algorithm's performance to the value of ϵ_r is small. MS-IBQN-LS has a similar sensitivity to ϵ_r in the approximations for the Jacobians of the flow solver and structural solver. In all other simulations for this first test case, ϵ_r is set to 10^{-10}.

Figure 7 shows the convergence of the normalized residual and the increase of the number of columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ during the coupling iterations in a representative time step with IQN-ILS and MS-IQN-ILS. A representative time step is defined as a time step which requires approximately the average number of coupling iterations per time step. Figure 8 depicts the same for IBQN-LS and MS-IBQN-LS. Figure 7a and Figure 8a clearly illustrate that the data recalculated by the additional solvers accelerates the convergence of the coupling iterations between $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$. In Figure 7a, it can be seen that the normalized residual is the same in the first and second coupling iteration of this particular time step for IQN-ILS and MS-IQN-ILS with $g = h = 3$. Of course, this is not always the case. The reason is that no differences from the previous time step had been recalculated by the additional solvers in the MS-IQN-ILS algorithm when $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ completed their first calculation. As a result $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ were empty at the beginning of the second coupling iteration and a relaxation step had to be performed, like in IQN-ILS. Figure 7b and Figure 8b show that the number of columns increases faster if more solvers are used; for the standard algorithms with one solver of each type, it is equal to the number of coupling iterations minus one.

5.2. Rolling tank

The second example is a rolling tank case, presented by Idelsohn et al. [38]. This case consists of a rectangular tank, filled with oil and air. An electric motor imposes a harmonic rolling motion of the tank around the midpoint of its bottom. Both fluids interact with the flexible beam which is clamped to bottom of the tank.

The oil and the air are both considered incompressible and mutually immiscible. This multi-phase flow is modelled with the volume-of-fluid (VOF) technique, which introduces a scalar volume fraction α_f throughout the fluid domain to distinguish the liquid from the gas [39, 40]. A region is filled with liquid only if the volume fraction is one and with gas only if the volume fraction is zero. The fluid properties such as the fluid density ρ_f are written as a function of the volume fraction

$$\rho_f = \alpha_f \rho_l + (1 - \alpha_f)\rho_g \tag{59}$$

with ρ_l and ρ_g the density of the liquid and the gas, respectively. A similar interpolation is used for the fluid viscosity μ_f. Gravity is the only body force so

$\vec{f}_f = -\rho_f g \vec{e}_y$ with $g = 9.81\,\mathrm{m/s^2}$ the gravitational acceleration and $\vec{e}_y$ the unit vector in the vertical direction as indicated in Figure 2. For two incompressible fluid phases, the mass conservation of the phases results in an equation for the volume fraction, namely

$$\frac{\partial \alpha_f}{\partial t} + \nabla \cdot (\alpha_f \vec{v}) = 0. \tag{60}$$

Neither mass transfer nor surface tension is taken into account between the liquid and the gas.

The flow equations are discretized in space on a grid with triangular and rectangular cells using the finite volume method. Scalars are stored in the cell centres and a power law is used to obtain momentum variables at the faces. Gradients at the cell centres are calculated from the face values using the Green-Gauss theorem. The face values for the gradient calculations are the arithmetic average of the node values, which are in turn the weighted average of the values in the cells around the node. The pressure interpolation at the faces is performed with a staggered grid approach similar to the one described by Patankar [41]. Eqs. (1) are solved using the pressure-implicit with splitting of operators (PISO) scheme with skewness and neighbour correction. Algebraic multi-grid (AMG) is employed to accelerate the convergence.

The grid of the fluid domain is deforming, driven by the deformation of the fluid-structure interface. Smoothing with fictitious springs between the grid nodes is applied for deformations during the time step. Cells which have either become too skewed or which fall outside the range of desired cell sizes are eliminated once in each time step. The implicit finite difference time discretization of Eqs. (1) in ALE formulation is first-order accurate on a moving grid. For the first calculation in a time step, the iterations in the flow solver begin from the flow field obtained in the previous time step. For all subsequent calculations in that same time step, the iterations in the flow solver begin from the flow field at the end of the previous calculation, which is often closer to the solution. As a result, fewer iterations inside the flow solver are required per calculation after the first calculation in each time step.

Eq. (60) for the volume fraction is solved with first-order explicit time discretization but the time step for this equation is only a fraction of the time step of the FSI calculation such that the Courant number does not exceed 0.25 near the liquid-gas interface. However, the volume fraction is recalculated after each solver iteration and the convective flux coefficients are updated based on the new volume fractions. The liquid-gas interface is reconstructed with a piecewise-linear

approach to obtain an accurate calculation of the fluxes through the faces near the liquid-gas interface [42].

The structure is discretized with rectangular 8-node continuum finite elements using reduced integration. Geometric nonlinearity is taken into account during the solution process and the stress on the fluid-structure interface follows the rotation of the structure in the time step. Unconditionally stable implicit Hilber-Hughes-Taylor time integration [43] is used with a small numerical damping parameter $\alpha_s = -0.05$. The constitutive equation of the isotropic beam material is a linear-elastic law with a plane stress approximation.

For this rolling tank, data from experiments (see Figure 9) and two-dimensional monolithic PFEM calculations are available [38]. The experiments have been performed with a transparent tank to be able to take images. The displacement of the tip of the beam in the rotating reference frame of the tank has been calculated from these images with a computer programme. Special attention has been paid to the gaps between the flexible beam and the front and back of the tank such that the experiments can be considered two-dimensional.

The tank is 0.609 m wide and 0.3445 m high; the oil level is 0.1148 m. The elastic beam is 0.004 m thick and its tip coincides with the still liquid-gas interface. The top of the tank is a constant pressure boundary while all other boundaries are no-slip walls. Figure 10 depicts the grid, which has 97840 degrees of freedom for the fluid and 1141 for the structure. Due to remeshing, the number of degrees of freedom in the fluid domain changes slightly during the simulations.

The angular frequency of the rolling motion that is imposed by the electric motor corresponds to the fundamental frequency of gravitational waves in a liquid of limited depth [44], given by

$$\Omega = \sqrt{\frac{\pi g}{L} \tanh \frac{\pi H}{L}} \tag{61}$$

with H the height of the liquid and L the width of the tank. The period of the rolling motion is thus 1.21 s and its amplitude is 4°. However, when the motor is started there is a transition from the initial rest state to the harmonic motion due to inertia and therefore the true time-angle curves [38] have been employed. The time step has been varied from 0.01 s to 0.0025 s. The finest time step corresponds to approximately 500 time steps in one period of the rolling motion. The properties of the liquid, gas and solid can be found in Table 1. The relaxation parameter ω is set to 10^{-2} and the tolerance for the detection of small diagonal elements ϵ_r is equal to 10^{-11} for all simulations of this second test case.

The shape of the deformed beam and the position of the liquid-gas interface using the smallest time step are compared with experimental data in Figure 9. For a more quantitative comparison, Figure 11 depicts the displacement of the tip of the beam parallel to the bottom of the tank (in the rotating reference frame). Good agreement can be observed between the experiments and the numerical results.

Table 2 lists the average number of coupling iterations between $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$ per time step for the different coupling algorithms and different time step sizes. The multi-solver algorithms use either 4 solvers of each type ($g = h = 4$) or 4 flow solvers and 2 structural solvers ($g = 4, h = 2$). The average number of coupling iterations decreases with 3 per time step by using the multi-solver version of both IQN-ILS and IBQN-LS. Table 2 also shows that the influence of the time step size on the performance of the presented algorithms is small for this case. For this more time-consuming case, the simulation is performed only once. In [31], it has been stated that this case cannot be simulated robustly with simple reuse (i.e. no recalculation) of data from previous time steps in the least-squares model. Here, it is shown that recalculating the differences from the previous time step at the current time level does accelerate the simulation.

Figure 12 depicts the normalized residual and the number of columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ during the coupling iterations in a representative time step. For MS-IQN-ILS with $g = h = 4$ (Figure 12a), the number of columns increases from 2 to 15 in only 4 coupling iterations. This suggests that each flow solver and structural solver does 4 calculations in this time step. However, for this rolling tank simulation, a flow calculation is significantly slower than a structural calculation. So, whenever a flow solver completes its calculation and puts the corresponding data in the intermediate queue, then one of the structural solvers quickly gets this data. While all flow solvers perform 4 calculations in this time step, significant differences are present between the structural solvers. $\boldsymbol{\mathcal{S}}_1$ performs 4 calculations, $\boldsymbol{\mathcal{S}}_2$ 10 calculations, $\boldsymbol{\mathcal{S}}_3$ 2 calculations and $\boldsymbol{\mathcal{S}}_4$ no calculations at all. As a flow calculation takes longer than a structural calculation, $\boldsymbol{\mathcal{S}}_2$ generally completes its calculation before new data are added to the intermediate FIFO queue by the flow solvers. Hence, $\boldsymbol{\mathcal{S}}_2$ is the first candidate to get the new data in the intermediate queue, except for 2 cases in which $\boldsymbol{\mathcal{S}}_3$ gets the new data from the intermediate queue because $\boldsymbol{\mathcal{S}}_2$ is still calculating. As can be seen in Table 2, a calculation with only 2 structural solvers results in approximately the same number of coupling iterations per time step.

The convergence of the residual is similar for MS-IBQN-LS as for MS-IQN-ILS, as can be seen in Figure 12b. However, there is a significant distinction in the number of columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$. In the MS-IBQN-LS algorithm, the model

for the Jacobian of the flow solver and for the Jacobian of the structural model are completely independent, so they do not have to contain the same number of columns. If $g = h = 4$, the additional structural solvers recalculate all differences from the previous time step during the first calculation between $\boldsymbol{\mathcal{F}}_1$ and $\boldsymbol{\mathcal{S}}_1$. As a result, the number of columns in the model for the structural solver jumps to 24 in the first coupling iteration. In each following coupling iteration, one difference, coming from $\boldsymbol{\mathcal{S}}_1$, is added to the structural model, while all other structural solvers wait.

5.3. Propagation of a pressure wave in a three-dimensional flexible tube

The third example is the propagation of a pressure wave in a three-dimensional flexible tube with radius 0.005 m and length 0.05 m, as described by Fernandez and Moubachir [45], Formaggia et al. [46], Gerbeau and Vidrascu [47]. This tube is a simplified model for a large artery. The same discretization and solution techniques as for the previous example are used, without the volume-of-fluid model. The fluid domain is divided into hexahedral cells and the structure is discretized using shell elements with 8 nodes. The grid contains 61440 degrees of freedom for the fluid and 1824 degrees of freedom for the structure.

The tube's wall is a linear elastic material with density 1200 kg/m^3, Young's modulus 3×10^5 N/m^2, Poisson's ratio 0.3 and thickness 0.001 m. The structure is clamped in all directions at the inlet and outlet. The fluid is incompressible and has a density of 1000 kg/m^3 and a viscosity of 0.003 Pas. Both the fluid and the structure are initially at rest. During the first 3×10^{-3} s, an overpressure of 1333.2 N/m^2 is applied at the inlet. The wave propagates through the entire tube during 10^{-2} s, simulated with 100 time steps. Pressure contours on the fluid-structure interface are shown in Figure 13 and they correspond well with those in [45–47].

The relaxation parameter ω is set to 10^{-2} and the tolerance for the detection of small diagonal elements ϵ_r is equal to 10^{-11} for all simulations of this example. The convergence criterion for the coupling iterations is $||\boldsymbol{r}_1^k||_2 < 10^{-3}||\boldsymbol{r}_1^0||_2$.

Table 3 lists the number of coupling iterations per time step, averaged over the entire simulation. The multi-solver algorithms use 4 solvers of each type ($g = h = 4$) as the duration of a flow calculation and a structural calculation is approximately the same. The average number of coupling iterations approximately halves by using MS-IQN-ILS instead of IQN-ILS. The MS-IBQN-LS results in a reduction of more than 40 % compared to IBQN-LS.

6. Conclusions

A new class of multi-solver quasi-Newton coupling techniques for partitioned simulation of unsteady fluid-structure interaction has been developed. These coupling techniques use more than one flow solver and more than one structural solver to solve a single coupled problem. Both the Multi-Solver IQN-ILS (MS-IQN-ILS) and the Multi-Solver IBQN-LS (MS-IBQN-LS) technique have been described in detail, starting from the existing coupling algorithms with one flow solver and one structural solver. These multi-solver techniques calculate the coupled solution with one flow solver and one structural solver and use the additional solvers to recalculate differences in the least-squares models from the previous time step. The numerical results show that these multi-solver algorithms can reduce the duration of an unsteady partitioned fluid-structure interaction simulation. The speed-up is independent of the speed-up due to parallellization of the solvers themselves and the algorithms can also be applied to accelerate solvers without parallellization. However, as the cost of the additional solvers is large, the multi-solver algorithms should only be used when the simulation cannot be accelerated further by increasing the number of cores per solver. Although the presented algorithms can be used to couple any pair of solvers, they are most suitable for black-box commercial codes.

Acknowledgments

J. Degroote gratefully acknowledges a postdoctoral fellowship of the Research Foundation - Flanders (FWO). The authors thank Antonio Souto-Iglesias from the Naval Architecture Department (ETSIN), Technical University of Madrid (UPM), for the experimental data of the rolling tank.

References

[1] T. Tezduyar, S. Sathe, K. Stein, Solution techniques for the fully discretized equations in computation of fluid-structure interactions with the space-time formulations, Computer Methods in Applied Mechanics and Engineering 195 (41–43) (2006) 5743–5753.

[2] M. Khan, M. Moatamedi, M. Souli, T. Zeguer, Multiphysics out of position airbag simulation, International Journal of Crashworthiness 13 (2) (2008) 159–166.

[3] K. Billah, R. Scanlan, Resonance, Tacoma Narrows Bridge Failure, and Undergraduate Physics Textbooks, American Journal of Physics 59 (2) (1991) 118–124.

[4] D. Dooms, Fluid-structure interaction applied to flexible silo constructions, Ph. D. thesis, Katholieke Universiteit Leuven, Department of Civil Engineering, 2008.

[5] C. Farhat, K. van der Zee, P. Geuzaine, Provably second-order time-accurate loosely-coupled solution algorithms for transient nonlinear computational aeroelasticity, Computer Methods in Applied Mechanics and Engineering 195 (17–18) (2006) 1973–2001.

[6] K. Willcox, J. Paduano, J. Peraire, Low Order Aerodynamic Models For Aeroelastic Control Of Turbomachines, in: 40^{th} AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, St Louis, MO, USA, 1–11, 1999.

[7] J.-F. Gerbeau, M. Vidrascu, P. Frey, Fluid-structure interaction in blood flows on geometries based on medical imaging, Computers & Structures 83 (2–3) (2005) 155–165.

[8] W. Wall, T. Rabczuk, Fluid-structure interaction in lower airways of CT-based lung geometries, International Journal for Numerical Methods in Fluids 57 (5) (2008) 653–675.

[9] C. Peskin, Numerical analysis of blood flow in the heart, Journal of Computational Physics 25 (3) (1977) 220–252.

[10] C. Peskin, Flow patterns around heart valves: a numerical method, Journal of Computational Physics 10 (2) (1972) 252–271.

[11] K. Dumont, J. Vierendeels, R. Kaminsky, G. Van Nooten, P. Verdonck, D. Bluestein, Comparison of the hemodynamic and thrombogenic performance of two bileaflet mechanical heart valves using a CFD/FSI model, Journal of Biomechanical Engineering - Transactions of the ASME 129 (4) (2007) 558–565.

[12] M. Astorino, J.-F. Gerbeau, O. Pantz, K.-F. Traore, Fluid-structure interaction and multi-body contact: Application to aortic valves, Computer Methods in Applied Mechanics and Engineering 198 (45–46) (2009) 3603–3612.

[13] K.-J. Bathe, H. Zhang, M. Wang, Finite element analysis of incompressible and compressible fluid flows with free surfaces and structural interactions, Computers & Structures 56 (2–3) (1995) 193–213.

[14] K.-J. Bathe, H. Zhang, Finite element developments for general fluid flows with structural interactions, International Journal for Numerical Methods in Engineering 60 (1) (2004) 213–232.

[15] M. Heil, An efficient solver for the fully coupled solution of large-displacement fluid-structure interaction problems, Computer Methods in Applied Mechanics and Engineering 193 (1–2) (2004) 1–23.

[16] J. Hron, S. Turek, A monolithic FEM/Multigrid Solver for ALE formulation of fluid structure interaction with application in biomechanics, in: H.-J. Bungartz, M. Schäfer (Eds.), Fluid-Structure Interaction – Modelling, Simulation, Optimisation, no. 53 in Lecture Notes in Computational Science and Engineering, Springer, Berlin, 146–170, 2006.

[17] S. Piperno, C. Farhat, B. Larrouturou, Partitioned procedures for the transient solution of coupled aeroelastic problems. Part I: model problem, theory and two-dimensional application, Computer Methods in Applied Mechanics and Engineering 124 (1–2) (1995) 79–112.

[18] M. Lesoinne, C. Farhat, A higher-order subiteration free staggered algorithm for non-linear transient aeroelastic problems, AIAA Journal 36 (9) (1998) 1754–1756.

[19] S. Piperno, C. Farhat, Partitioned procedures for the transient solution of coupled aeroelastic problems. Part II: energy transfer analysis and three-dimensional applications, Computer Methods in Applied Mechanics and Engineering 190 (24–25) (2001) 3147–3170.

[20] A. van Zuijlen, A. de Boer, H. Bijl, Higher-order time integration through smooth mesh deformation for 3D fluid-structure interaction simulations, Journal of Computational Physics 224 (1) (2007) 414–430.

[21] E. van Brummelen, Added mass effects of compressible and incompressible flows in fluid-structure interaction, Journal of Applied Mechanics 76 (2) (2009) 021206–1–7.

[22] P. Causin, J.-F. Gerbeau, F. Nobile, Added-mass effect in the design of partitioned algorithms for fluid-structure problems, Computer Methods in Applied Mechanics and Engineering 194 (42–44) (2005) 4506–4527.

[23] C. Förster, W. Wall, E. Ramm, Artificial added mass instabilities in sequential staggered coupling of nonlinear structures and incompressible viscous flows, Computer Methods in Applied Mechanics and Engineering 196 (7) (2007) 1278–1293.

[24] D. Mok, W. Wall, E. Ramm, Accelerated iterative substructuring schemes for instationary fluid-structure interaction, in: K.-J. Bathe (Ed.), Computational Fluid and Solid Mechanics, Elsevier, 1325–1328, 2001.

[25] U. Küttler, W. Wall, Fixed-point fluid-structure interaction solvers with dynamic relaxation, Computational Mechanics 43 (1) (2008) 61–72.

[26] C. Michler, E. van Brummelen, R. de Borst, An interface Newton-Krylov solver for fluid-structure interaction, International Journal for Numerical Methods in Fluids 47 (10-11) (2005) 1189–1195.

[27] J. Degroote, K.-J. Bathe, J. Vierendeels, Performance of a new partitioned procedure versus a monolithic procedure in fluid-structure interaction, Computers & Structures 87 (11–12) (2009) 793–801.

[28] J. Vierendeels, L. Lanoye, J. Degroote, P. Verdonck, Implicit coupling of partitioned fluid-structure interaction problems with reduced order models, Computers & Structures 85 (11–14) (2007) 970–976.

[29] S. Badia, F. Nobile, C. Vergara, Robin-Robin preconditioned Krylov methods for fluid-structure interaction problems, Computer Methods in Applied Mechanics and Engineering 198 (33–36) (2009) 2768–2784.

[30] J. Degroote, R. Haelterman, S. Annerel, P. Bruggeman, J. Vierendeels, Performance of partitioned procedures in fluid-structure interaction, Computers & Structures 88 (7–8) (2010) 446–457.

[31] J. Degroote, A. Souto-Iglesias, W. Van Paepegem, S. Annerel, P. Bruggeman, J. Vierendeels, Partitioned simulation of the interaction between an elastic structure and free surface flow, Computer Methods in Applied Mechanics and Engineering 199 (33–36) (2010) 2085–2098.

[32] C. Farhat, M. Lesoinne, P. Le Tallec, Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: momentum and energy conservation, optimal discretization and application to aeroelasticity, Computer Methods in Applied Mechanics and Engineering 157 (1–2) (1998) 95–114.

[33] G. Golub, C. Van Loan, Matrix computations, Johns Hopkins University Press, Baltimore, MD, USA, 3rd edn., 1996.

[34] J. Degroote, P. Bruggeman, R. Haelterman, J. Vierendeels, Stability of a coupling technique for partitioned solvers in FSI applications, Computers & Structures 86 (23–24) (2008) 2224–2234.

[35] J. Degroote, S. Annerel, J. Vierendeels, Stability analysis of Gauss-Seidel iterations in a partitioned simulation of fluid-structure interaction, Computers & Structures 88 (5–6) (2010) 263–270.

[36] Y. Saad, M. Schultz, GMRES: A Generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal on Scientific & Statistical Computing 7 (3) (1986) 856–869.

[37] J. Vierendeels, K. Riemslagh, E. Dick, A multigrid semi-implicit line-method for viscous incompressible and low-Mach-number flows on high aspect ratio grids, Journal of Computational Physics 154 (2) (1999) 310–341.

[38] S. Idelsohn, J. Marti, A. Souto-Iglesias, E. Oñate, Interaction between an elastic structure and free-surface flows: experimental versus numerical comparisons using the PFEM, Computational Mechanics 43 (1) (2008) 125–132.

[39] C. Hirt, B. Nichols, Volume of fluid method for the dynamics of free boundaries, Journal of Computational Physics 39 (1) (1981) 201–225.

[40] W. Rider, D. Kothe, Reconstructing Volume Tracking, Journal of Computational Physics 141 (2) (1998) 112–152.

[41] S. Patankar, Numerical Heat Transfer and Fluid Flow, Hemisphere, Washington DC, USA, 1980.

[42] D. Youngs, Time-Dependent Multi-Material Flow with Large Fluid Distortion, in: K. Morton, M. Baines (Eds.), Numerical Methods for Fluid Dynamics, Academic Press, New York, NY, USA, 273–285, 1982.

[43] H. Hilber, T. Hughes, R. Taylor, Improved numerical dissipation for time integration algorithms in structural dynamics, Earthquake Engineering & Structural Dynamics 5 (3) (1977) 283–292.

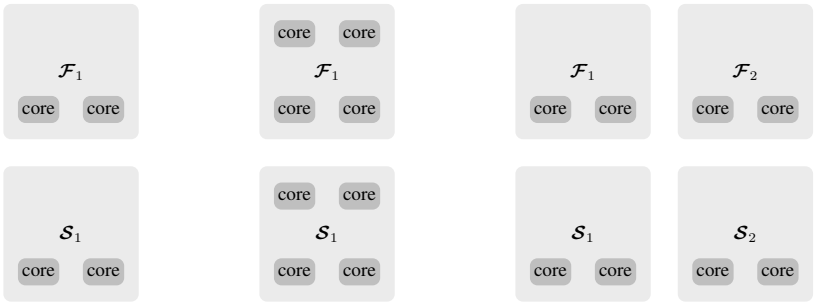
[44] H. Lamb, Hydrodynamics, Cambridge University Press, 6th edn., 1932.

[45] M. Fernandez, M. Moubachir, A Newton method using exact Jacobians for solving fluid-structure coupling, Computers & Structures 83 (2–3) (2005) 127–142.

[46] L. Formaggia, J.-F. Gerbeau, F. Nobile, A. Quarteroni, On the coupling of 3D and 1D Navier-Stokes equations for flow problems in compliant vessels, Computer Methods in Applied Mechanics and Engineering 191 (6–7) (2001) 561–582.

[47] J.-F. Gerbeau, M. Vidrascu, A quasi-Newton algorithm based on a reduced model for fluid-structure interaction problems in blood flows, ESAIM: Mathematical Modelling and Numerical Analysis 37 (4) (2003) 631–648.

7. Figures



(a) (b) (c)

Figure 1: In (a), the flow solver $\mathcal{F}_1$ and structural solver $\mathcal{S}_1$ each run on two cores. With twice as many cores, either (b) the number of cores per solver can be increased to four or (c) an additional flow solver $\mathcal{F}_2$ and structural solver $\mathcal{S}_2$ can be started.

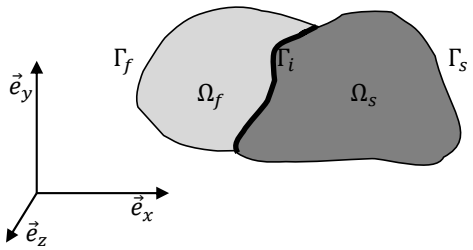


Figure 2: The abstract representation of the fluid subdomain Ω_f, the structure subdomain Ω_s, their boundaries Γ_f and Γ_s and the fluid-structure interface Γ_i.

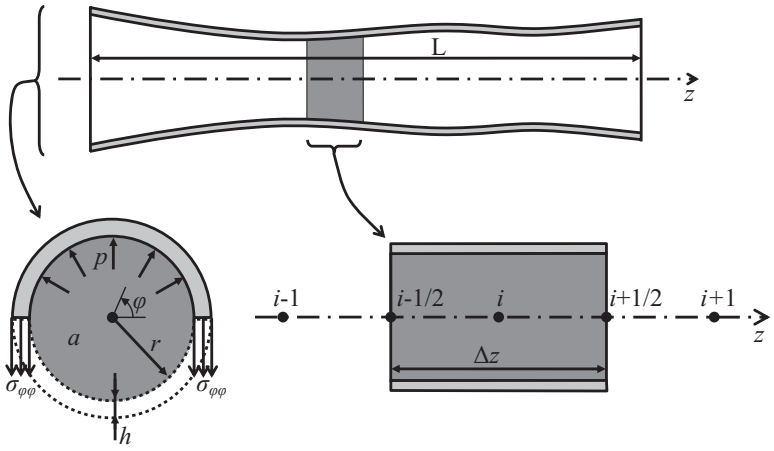


Figure 3: The model for flexible tube with details of the cross-section and a control volume used in the discretization of the governing equations.

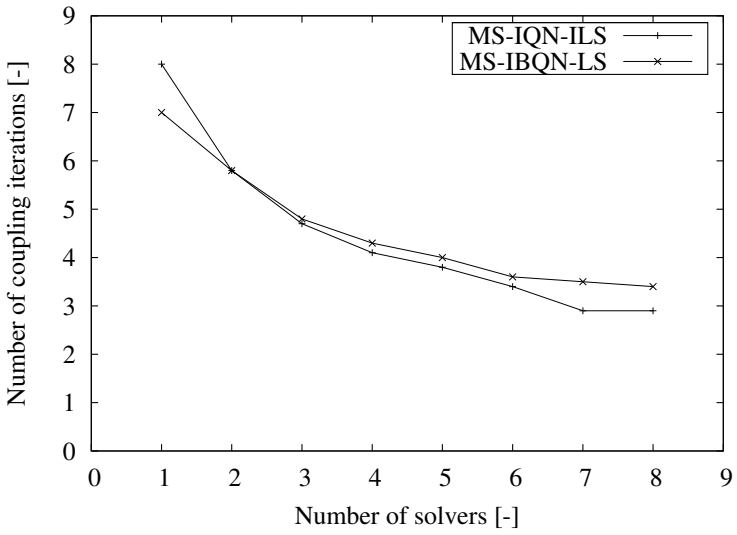


Figure 4: The average number of coupling iterations per time step as a function of the number of solvers ($g = h$) for the flexible tube.

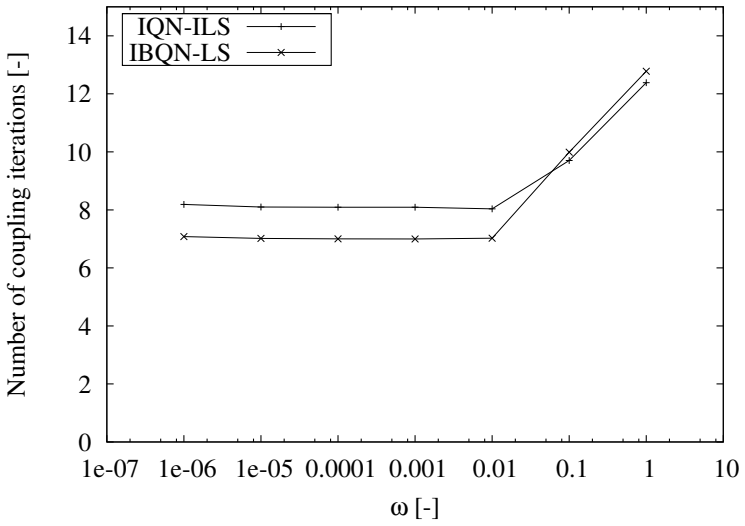


Figure 5: The average number of coupling iterations per time step as a function of ω for the flexible tube ($g = h = 1$).

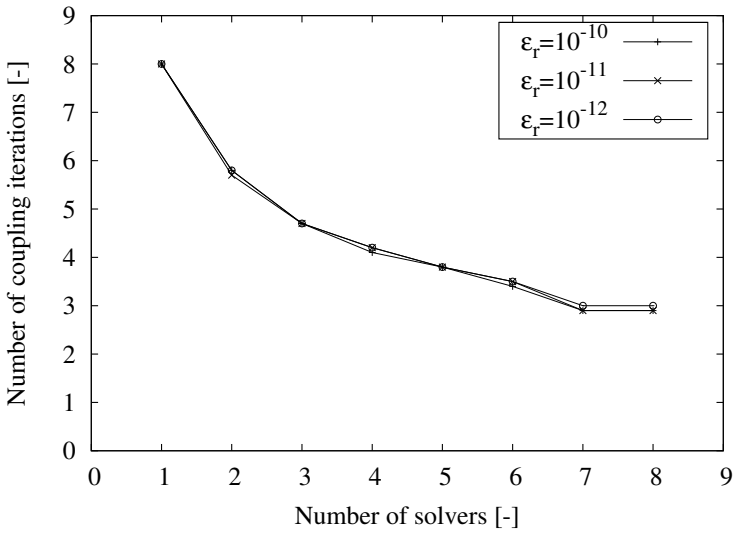
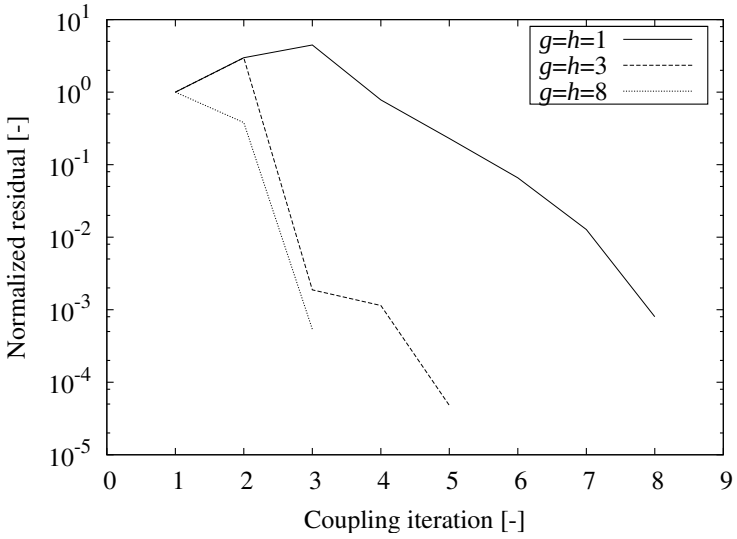
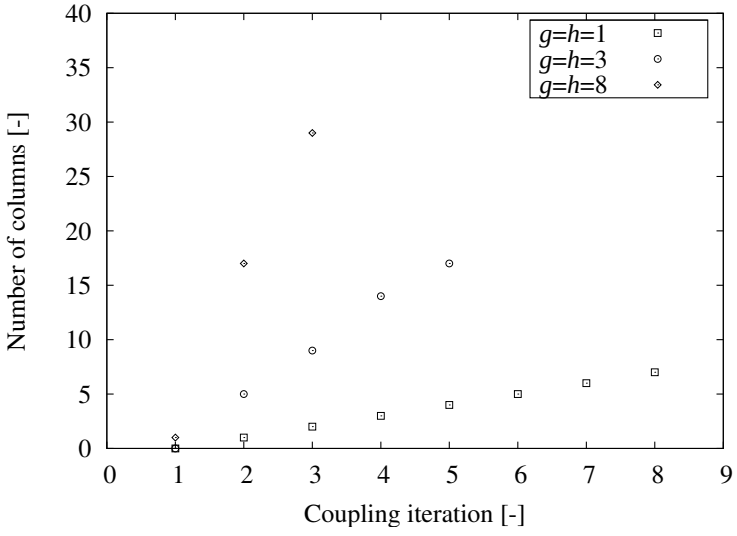


Figure 6: The average number of coupling iterations per time step as a function of the number of solvers ($g = h$) for the flexible tube using MS-IQN-ILS and different values of ϵ_r.

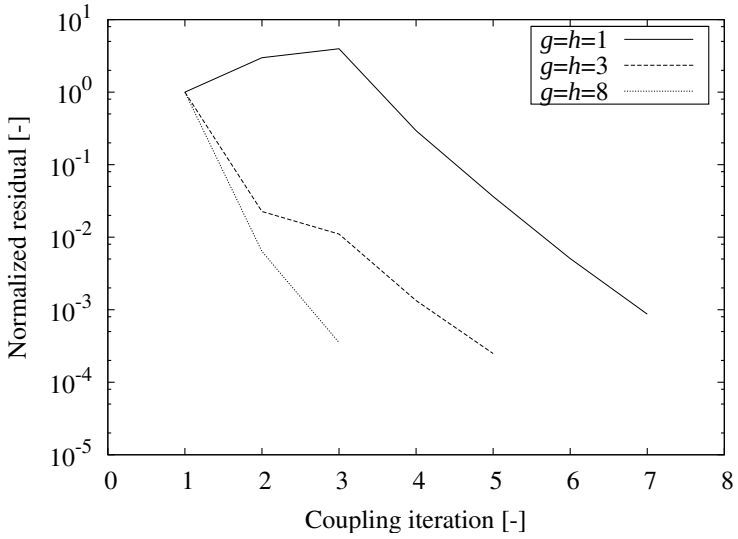


(a)

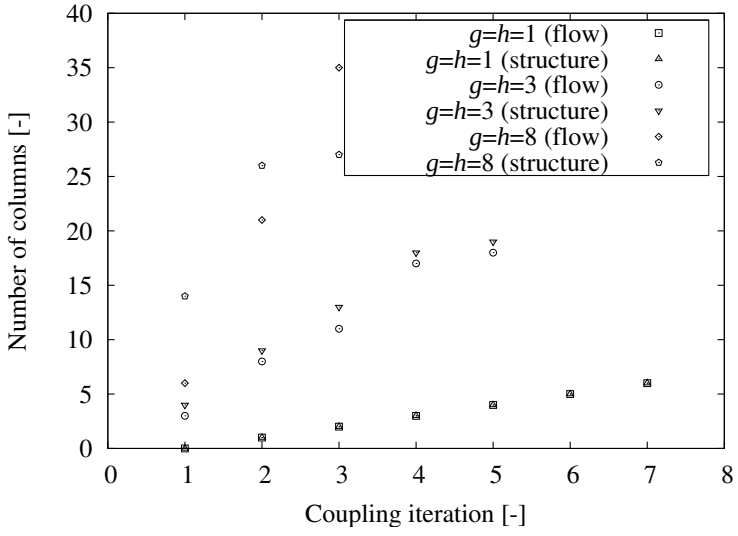


(b)

Figure 7: (a) The normalized residual ($||\boldsymbol{r}_1^k||_2/||\boldsymbol{r}_1^0||_2$) and (b) the number of columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ during the coupling iterations in a representative time step for the flexible tube, using IQN-ILS and MS-IQN-ILS with 3 and 8 solvers of each type.



(a)



(b)

Figure 8: (a) The normalized residual ($||\boldsymbol{r}_1^k||_2/||\boldsymbol{r}_1^0||_2$) and (b) the number of columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ during the coupling iterations in a representative time step for the flexible tube, using IBQN-LS and MS-IBQN-LS with 3 and 8 solvers of each type.

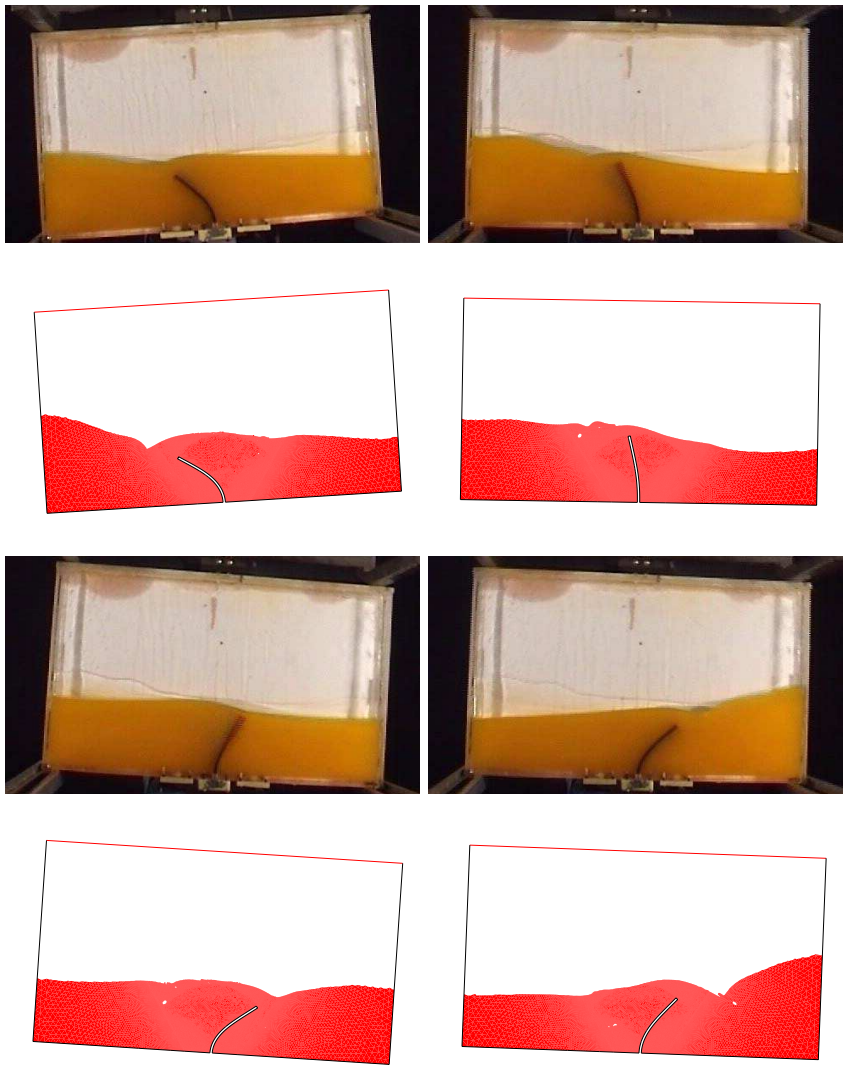


Figure 9: The comparison between the experimental data and the numerical results for the rolling tank after 1.84 s, 2.12 s, 2.32 s and 2.56 s with $\Delta t = 0.0025$ s.

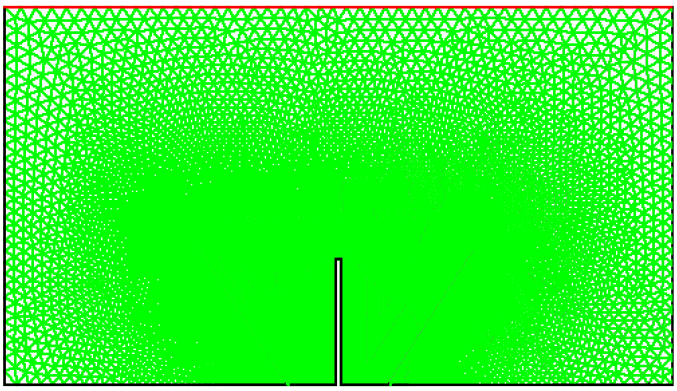


Figure 10: The grid for the simulation of the rolling tank. A constant pressure is imposed on the red boundary, green is the fluid domain and black represents a no-slip wall or the structural domain.

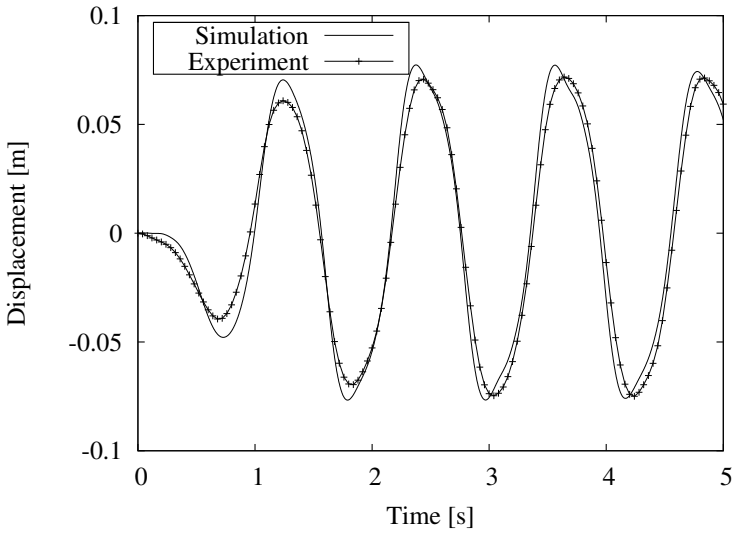


Figure 11: The displacement of the tip of the beam parallel to the bottom of the tank (in the rotating reference frame) for the simulation of the rolling tank with $\Delta t = 0.0025\,\mathrm{s}$.

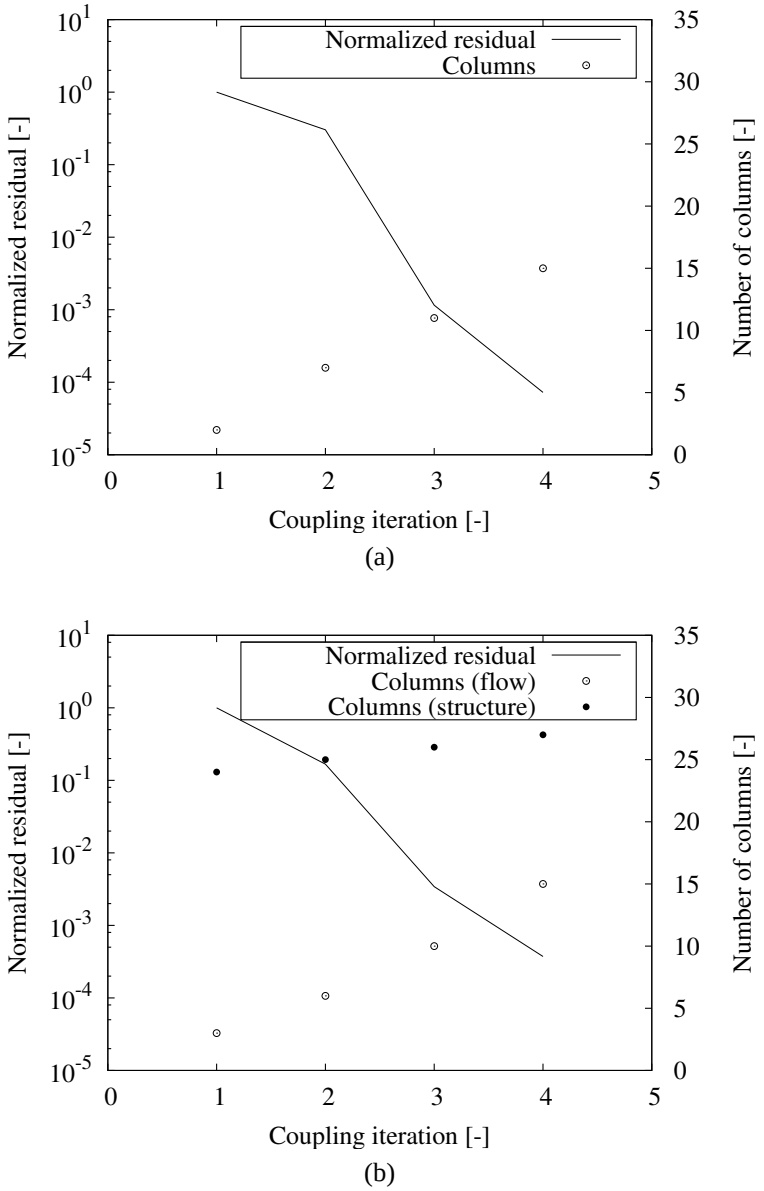


Figure 12: The normalized residual ($||\boldsymbol{r}_1^k||_2/||\boldsymbol{r}_1^0||_2$) and the number of columns in $\boldsymbol{V}^k$ and $\boldsymbol{W}^k$ during the coupling iterations in a representative time step for the rolling tank, using (a) MS-IQN-ILS and (b) MS-IBQN-LS, both with 4 solvers of each type ($g = h = 4$) and $\Delta t = 0.0025$ s.

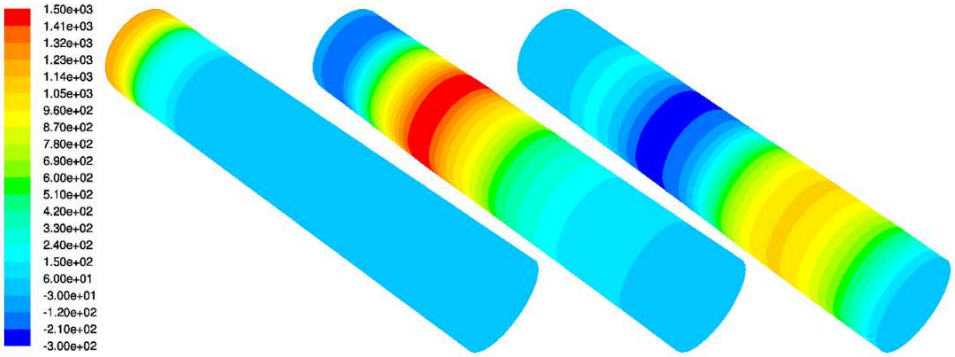


Figure 13: The pressure contours (in Pa) on the fluid-structure interface for the propagation of a pressure wave in a three-dimensional flexible tube after 10^{-3} s (left), 5×10^{-3} s (centre) and 9×10^{-3} s (right).

8. Tables

Liquid	ρ	917	kg/m^3
	μ	0.04585	Pas
Gas	ρ	1.225	kg/m^3
	μ	1.79×10^{-5}	Pas
Solid	ρ	1100	kg/m^3
	E	6×10^{6}	N/m^2
	ν	0.49	

Table 1: The parameter values that are used to model the rolling tank.

Algorithm	g	h	Δt [s]	Iterations
IQN-ILS	1	1	0.0025	7.8
IQN-ILS	1	1	0.005	7.9
IQN-ILS	1	1	0.01	7.8
MS-IQN-ILS	4	2	0.0025	5.0
MS-IQN-ILS	4	4	0.0025	4.8
MS-IQN-ILS	4	4	0.005	4.5
MS-IQN-ILS	4	4	0.01	4.7
IBQN-LS	1	1	0.0025	7.3
MS-IBQN-LS	4	2	0.0025	4.2
MS-IBQN-LS	4	4	0.0025	4.1

Table 2: The average number of coupling iterations per time step for the simulation of the rolling tank. Different combinations of the number of solvers and the time step size are shown.

Algorithm	g	h	Iterations
IQN-ILS	1	1	12.4
MS-IQN-ILS	4	4	6.5
IBQN-LS	1	1	12.8
MS-IBQN-LS	4	4	7.3

Table 3: The average number of coupling iterations per time step for the simulation of the propagation of a pressure wave in a three-dimensional flexible tube.